Developing a Client/Server Architecture for a Mobile AR Urban Design Application

---

A thesis submitted in partial fulfilment of the requirements for the Degree

of Masters in Human Interface Technology

in the University of Canterbury

by Michael Partridge

University of Canterbury

2013

Developing a Client/Server Architecture for a Mobile AR Urban Design Application

Michael Partridge

07-July-2013

Abstract

This thesis describes research into developing a client/server architecture for a mobile Augmented Reality (AR) application. Following the earthquakes that have rocked Christchurch the city is now changed forever. CityViewAR is an existing mobile AR application designed to show how the city used to look before the earthquakes. In CityViewAR 3D virtual building models are overlaid onto video captured by a smartphone camera. However the current version of CityViewAR only allows users to browse information stored on the mobile device. In this research the author extends the CityViewAR application to a client-server model so that anyone can upload models and annotations to a server and have this information viewable on any smartphone running the application. In this thesis we describe related work on AR browser architectures, the system we developed, a user evaluation of the prototype system and directions for future work.

Contents

1 Introduction

On September 4th 2010, a magnitude 7.1 earthquake hit the city of Christchurch in New Zealand and changed it forever. Since that time more than 11,000 aftershocks have shaken the city and over 1300 inner city buildings have been demolished. With nearly a third of downtown Christchurch gone it is difficult for people to remember what the city looked like before the earthquake hit, or what is might be like in the future.

However, Augmented Reality (AR) technology can be used to go back in time and see the city as it was, and also show future city designs. AR is technology that allows virtual images to be seamlessly included into views of the real world. Over the last year staff and students of the Human Interface Technology Laboratory New Zealand (HIT Lab NZ) have developed a mobile AR application, CityViewAR [13] [5], which allows users to see 3D virtual models of buildings put back on the real sites that they used to occupy, see Fig. 1. CityViewAR can show geo-located information using various visualization methods, including an interactive digital map, immersive 360 degree photography, and Augmented Reality (AR).

Figure 1: CityViewAR application showing a virtual building in place.

The current version of CityViewAR shows the city as it was the day before the earthquake and right after. However it is a stand-alone application with all the 3D model content downloaded once when the application is first installed. This means that it is difficult to upgrade the content with new models, and that it doesn't allow users to share their own content with others. This project will upgrade the CityViewAR application to allow users to share content with other users over the Internet, and for the material to be easily added through a web-based interface.

Thus the main goal of this project is:

1. To develop a client/server architecture for the CityViewAR application
2. To use this to extend the application to better support Urban Design.

The main capability that will be added is the ability for developers to easily add virtual content through a web-based interface without the need for programming so that end users can view it on the mobile AR application. This is important because the average user does not know how to program.

Developing this client/server interface will make the program more accessible to the general user and allow anyone to place virtual buildings or annotations in the real world. This change will also make CityViewAR the application appeal to a wider audience, and also enable it to be used by cities other than Christchurch.

Allowing anyone to upload 3D models into the CityViewAR application will allow professionals such as architects and urban designers to show what buildings or other structures will look like in a given space. Seeing the 3D model of a structure will complement current methods of showing what a structure looks like as the surrounding structures will also be visible.

Social aspects of the program will also be important. Allowing people to tag locations will give the program more of an appeal to the regular user.

Once virtual models of proposed buildings are shown in the real world people could tag the locations to give feedback on the planned content. In this way members of the community will be able to give feedback on proposed urban designs.

Compared to the existing CityViewAR application, this thesis will make the following contributions:

- Developing a Client/Server architecture so that objects are retrieved from a server rather than stored on device.
- Developing an intuitive authoring tool to allow people to add their own 3D buildings or structure of urban design to the application.
- Allow developers to create private channels to display their objects.

In order to do this, the following research issues must be addressed:

- Client/Server architecture
- User interface design of the web-based authoring tool
- Focus towards Urban Design structure of both interfaces
- Evaluation of the interfaces developed and the user experience

2 Background and Related Research

Augmented Reality involves the overlay of virtual content on views of the real world. In recent years mobile AR has started to become very popular. With the rise of faster and faster hardware in smartphones, and integrated sensors, it is becoming easier to add AR features into mobile applications. One of the first mobile AR prototypes was the Touring Machine, shown in Fig. 2, developed by Steve Feiner [2]. This combined a backpack, a head mounted display, and a tablet interaction to allow users to see and interact with virtual labels added to the real buildings around them. This large device was able to display a text overlay onto the real world and included a menu design to allow options to be selected. In addition to the Touring Machine, a number of other research groups experimented with backpack based mobile AR systems with applications such as AR Quake [16], and the HIT Lab NZ Outdoor AR Surveying project [7], among others.

Figure 2: The Touring Machine

Since then mobile AR has moved on to the cell phone. One of the early cell phone mobile AR projects was the Nokia Mara [8]. Like the Touring machine it was able to display simple text overlay onto the real world, see Fig. 3. To enable the Nokia phone to do this an external box containing GPS and compass sensors was needed, shown in Fig. 4. This made it unwieldy for a phone, but it was still a step up from the Touring Machine.

Figure 3: Nokia Mara screen output [8]

These days mobile AR is far more powerful and can incorporate information from all the smartphone sensors to overlay complex graphics onto real life, without need for an external sensor box. Commercial applications such as Junaio [14], Layar [12], or Wikitude [20] allow millions of people to see virtual content in the world around them on their own handheld devices.

There are two main ways to present virtual content in mobile AR applications. The first is to store all objects on the phone and use the phone sensor information to pick which ones to display. The disadvantage of this method is that it is harder to add new models to the scene and can take a large

Figure 4: Nokia Mara with sensor box

amount of storage space on the phone. The second is to use a client/server architecture where all the models are stored on a central server. Using information from its sensors the smartphone requests the relevant objects and sends them to the mobile AR client as needed. The disadvantage of this method is the requirement for network connectivity and possible costly data usage charges. Since the existing CityViewAR application doesn't currently have a client/server architecture, the focus of our work will be on the second approach.

Thus, this work will involve research and development in three areas:

1. Client/Server architecture for mobile AR.
2. A web based user interface for adding content to the AR scene.
3. User evaluation of the web interface.

2.1 Client-Server Architectures for Mobile AR

As stated above, a client/server architecture will be used in this project so that information can be stored on a server and the client can access as needed. In mobile AR, 3D models and other content will be stored on the server and a portion of this information will be downloaded onto the smartphone. To display the information, the smartphone makes a request to the server for the relevant information which is then supplied to be shown on the screen.

Currently there are multiple applications for smartphones that use a client/server interface to display AR information. These all send information from the phone to a server, which returns the correct information back to the phone. Fig. 5. shows the general process. Mobile AR applications that use this approach include AR Browsers which use a mobile AR client to show whatever content the web service provides them.

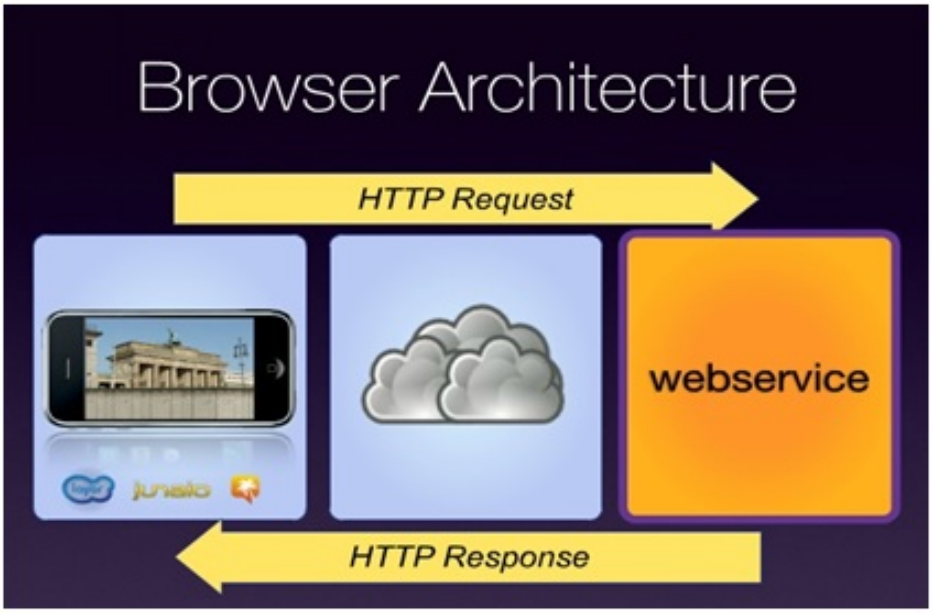


Figure 5: General client/server approach for mobile AR

Two of the most popular mobile AR browsers are Layar [12] and Junaio

[14], with tens of millions of users and thousands of AR information channels. Junaio and Layar support both location-based and computer vision based AR. The location-based AR uses GPS and orientation data from the phone to place virtual content onto live views of real world locations. Computer vision based AR matches the image from the camera to a database and is able to do real time image tracking and pose calculation so that virtual content can be overlaid on the printed image. This is widely used to add advertisements and other information to print media.

In our work we are going to focus just on location based AR, since this is the most suitable for outdoor urban design application. For this approach, Layar uses the client/server architecture shown in Fig. 6.

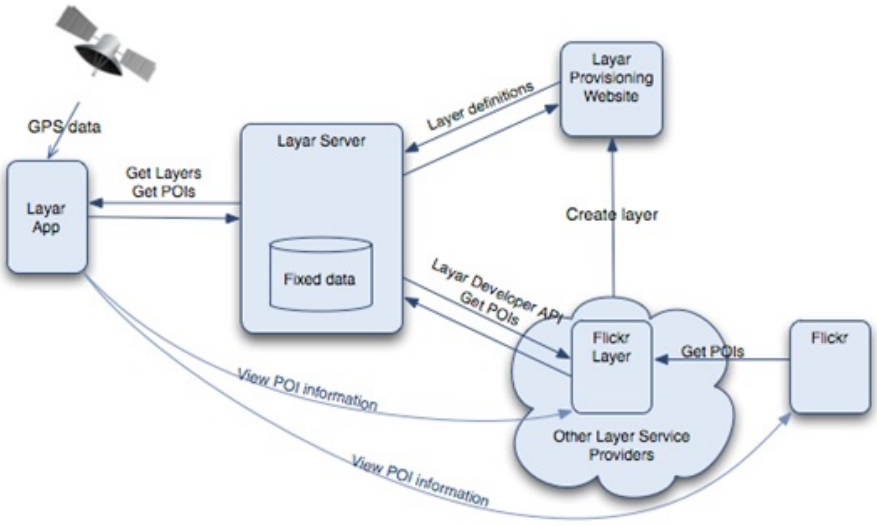


Figure 6: Layar Architecture [12]

A basic overview of how the Layar Architecture works is:

1. A User launches the Layar Reality Browser on a supported mobile device.

2. The Layar Client will send a request to the Layar Server.
3. Based on the request, the Layar Server will retrieve layer definitions from the publishing website.
4. A list of retrieved layers will be sent by the Layar Server and displayed on the Layar Client.
5. The User launches an AR content layer from this list
6. A getPOIs request is sent to the Layar Server.
7. The Layar Server forwards the Layer Service Provider of that layer.
8. The layer Service Provider returns AR content based on the Developer API (getPOIs response) back to the Layar Server.
9. The Layar Server validates the getPOIs response and sends it back to the Layar Client.
10. The Layar Client displays the getPOIs response to the User.

Another client/server architecture for mobile AR is seen with MobAR [15] being developed by the Open Mobile Alliance. This architecture, seen in Fig. 7, has the server calculating the client's location using information from the cellular provider. Information is then pushed onto the client by the server. Content providers can add POIs to the server, but the client is unable to. This architecture has the disadvantages of receiving the location from the cellular provider and not allowing the client to add POIs to the server. Therefore this architecture is unsuitable to our project.

The HIT Lab NZ has developed the Outdoor AR SDK which is a library primarily concerned with developing location based AR experiences on mobile devices. The goal of this thesis work will be to create a client/server version of the Outdoor AR SDK similar to the Layar architecture. This will

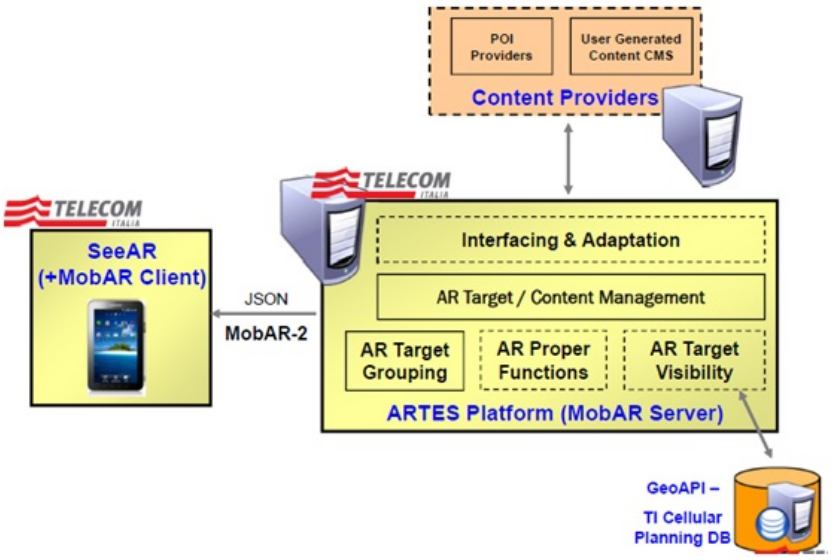


Figure 7: MobAR Architecture [15]

contain two main libraries: One each for the server and client. In addition an authoring will be developed tool for adding content to the AR server. This is likely the best approach for us to use as it maintained by the HIT Lab and therefore easily available.

2.2 Easy Authoring Tools for Mobile AR

In order to be able to add content to the server in a client/server mobile AR application, easy to use authoring tools are available. These include BuildAR [9], Layar Creator [12], BirdsView [1], and Hoppala [6]. In general they allow the user to drop text and other annotations onto an online map. Content placed on the map is then published to the server, and anyone with the correct AR browser application on their smartphone will be able to see it.

Birdsview primarily deals with overlaying text and images over real world locations and can create server content for either the Junaio or Layar platforms. The web interface contains the map shown in Fig. 8. The user can

then create content at a specific location by clicking on the desired location and uploading content. The allowable types of content include:

- Text.
- Images.
- Web addresses.

Figure 8: BirdsView content adding map [1]

Using GPS location data and orientation from the phone the content can then be seen in an AR view on the Junaio or Layar browsers. Fig. 9. shows the tagged locations in the cameras view site. Users can then select the tags to get extra information about the location. Users can also add their own tags using the phone or through the internet site. In this section we describe each of these in more detail.

BuildAR allows the user to upload multiple media types to its servers organised into Points of Interest (POI). The POIs can be placed using an address or longitude and latitude co-ordinates and once placed can be easily

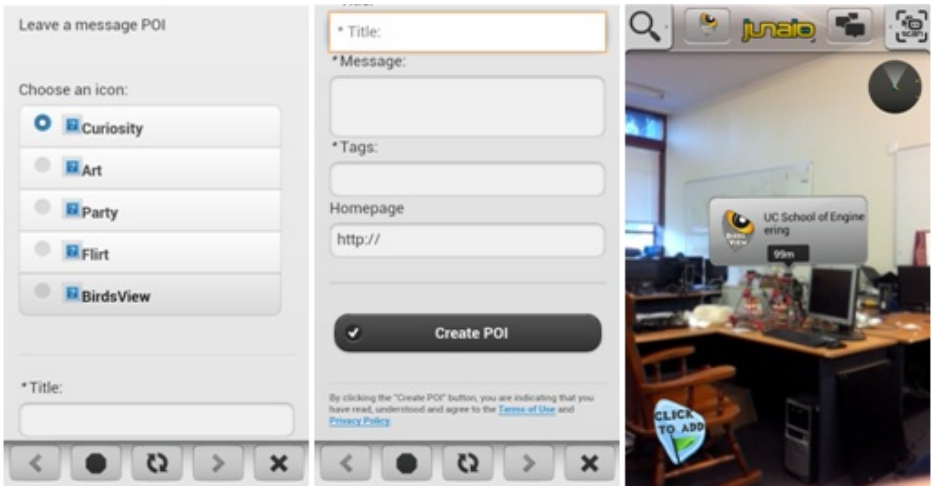


Figure 9: BirdsView POI creator and a finished tag

dragged to move to a new location. Each POI can contain multiple media types that are named projections and can include:

- Images.
- Videos.
- 3D models.

These can be moved, rotated, and scaled relative to the location of the POI. The downside to this system is that only the location of the POI is shown on the map. No information about the POI is shown. This can make it confusing to place the locations where the user wants.

Unfortunately, the user interface of the BuildAR website is slow to load and confusing to use. For example, when selecting to view a POI on the map every POI is shown which can make it hard to distinguish the one the user wants from the others. Leaving the map view is also difficult, requiring the user to select a button that is difficult to find. As the website is still in beta this will likely change to be more user friendly.

BuildAR and Layar allow the user to upload visual tracking images online. Once an image is uploaded the user can then add annotations, web links, and

video that will be played when someone scans the image in the application. These examples are all user friendly and require no programming skill to create. The Layar Creator, shown in Fig. 10, allows the user to upload an image to the website and then overlay them with buttons and other content to the image. Once this is done the user can publish them instantly. This type of tracking does not use any of the positional sensors and instead use video from the camera to track objects. So this means that is may not be suitable for outdoor AR applications.

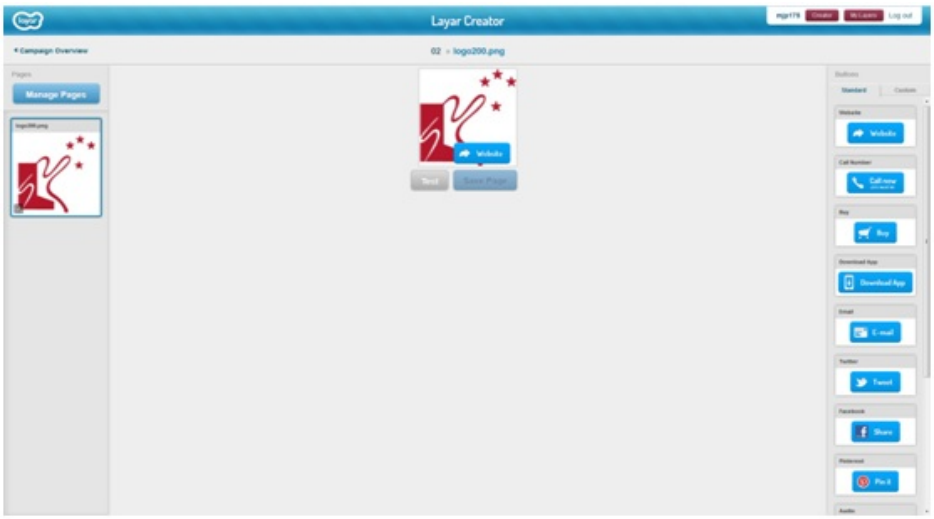


Figure 10: Layar Creator interface showing buttons that can be added

2.3 Content Creation on the Mobile AR Device

In addition to authoring mobile AR content through a website, the content can also be created on the mobile device itself. Three examples that allow in-app authoring of information are BirdsView [1], Sekai Camera [19], and TagWhat [17]. These applications show the user virtual tags on the screen above nearby locations and allow them to add their own tags, see Fig. 9. The tags shown are simple with text and images. Adding the tags is easy to do with the user using touch screen input to add some text about their current

location that is uploaded to the servers. All three examples are similar in function and only differ in interface design.

AR annotations can also be added to capture image content. For example, in Langlotz [11] the user takes a panorama of their current location that they can then annotate with text or voice. The photo with annotation is then tagged with their GPS coordinate and uploaded to a server. This was further extended to allow the user to add annotations onto the ground frame using GPS, but also allows the user to add 3D content directly from the phone [10], shown in Fig. 11. It does this by freezing the image that the camera takes and allowing the user to place basic primitives or copy real world objects. This data is all tagged with the GPS data and sent to a server so that other people may see it. The ability to limit who can see the objects and annotations is also included.

Figure 11: Langlotz interface copying a building

These all show features that we will want in our finished product. However the ability to add 3D objects from the phone is more than what is required. Neither of these allows fast authoring of material as the location of the annotations must be selected rather than just using the GPS data. Also neither has an easy online authoring tool associated with them. Our research will allow complex authoring from a web based desktop tool.

In summary, there are a number of tools for allowing non-programmers to upload content to servers for later retrieval. However these have a number of limitations, including:

- Confusing interfaces.
- Imprecise placement of objects.
- Slow to author items.

In our research we want to develop an authoring tool that overcomes these limitations. In particular we want to develop a tool that allows the user to:

- Create content.
- Edit created content.
- Do this without a confusing interface

In the next chapter we describe how we designed our prototype interface, and then in chapter four we present the interface implementation.

3 Design

The design process was performed with input from developers of the mobile CityViewAR application and the developer of the server architecture. Weekly meetings were scheduled where the current status of the server, web interface, and mobile application were discussed. The following week after the meeting would be developing the interface and implementing suggestions from the other developers.

3.1 Architecture

How the web interface and mobile client would interact with the server was the first thing decided. As they both needed to access the same information a simple architecture that both could handle needed to be chosen. The Representational state transfer (REST) API was chosen to do this as described later in this thesis.

Fig. 12. shows the final architecture chosen for the client-server architecture. The REST API would sit on top of the server. Any incoming requests would be handled by this. To access the server via a web browser the web interface described in this thesis would be used. Applications on mobile clients would used the OutdoorAR data manager to access the data from the server. This architecture was chosen so that the server didn't need to work out where the requests were coming from and could serve either device.

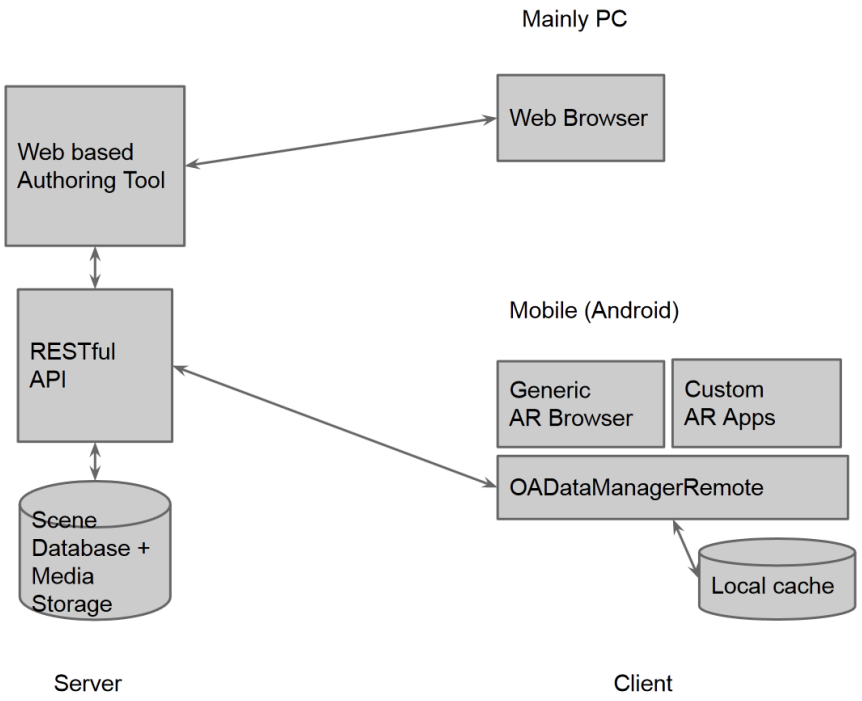


Figure 12: Design architecture

3.2 Schema

Once the overall architecture was decided the next step was to design how data would be stored and therefore accessed on the server. Fig. 13. shows the chosen schematics. The lowest level of the schematics are the models and media. Once these are on the server, instances of them would be linked to specific scenes. This was so that a single model or piece of media could be used across multiple scenes.

A channel is a collection made up of many related scenes. For example the user could specify a channel for post earthquake Christchurch. This channel could then be filled with scenes containing pre-earthquake models. That channel has a category that is passed down to the scenes that it is composed

of. The channel contains information about which users are able to view it. If the user is able to view the channel separate information specifies if the user is able to edit the channel and the lower levels it contains. This is so that the non authorised users can still view the channel.

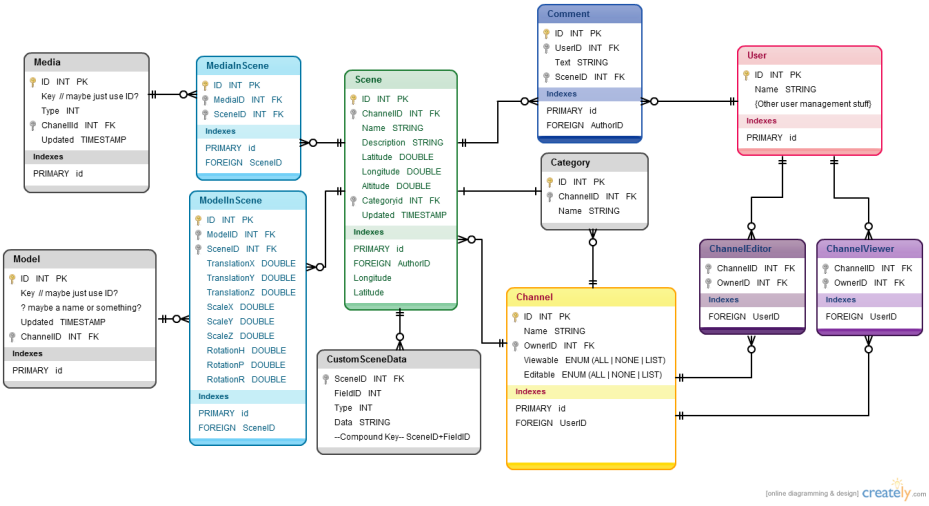


Figure 13: Server schematics

3.3 Page Design

Several sketches of potential designs of the website were made, one of which can be seen in Figs. 15. and 14. This shows an early mock-up of the website. The mock-up Scene page shows a button to switch between a map and list view. The map view featured a small list with the names of the scenes, while the list view contained a lot of information on the scene, such as:

- Name of the Scene.
- Central location in latitude and longitude co-ordinates.
- Edit and delete buttons.

The models and media are shown with a scrollable view and editing them is done without seeing them located on a map..

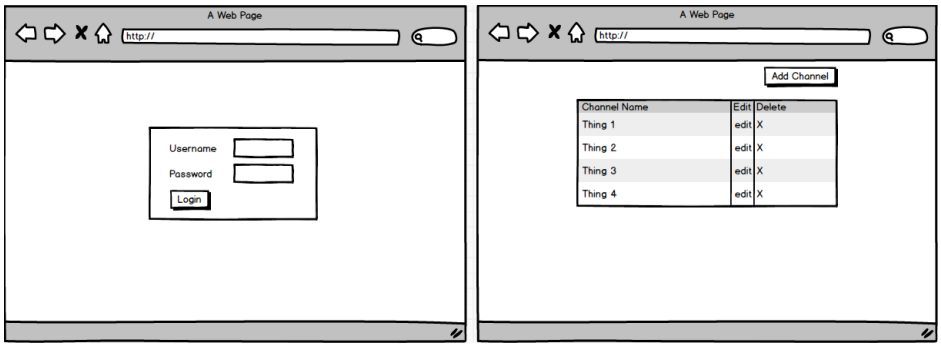


Figure 14: OutdoorAR authoring UI mock-up. Shows login and channel page

3.3.1 Login Page

The login page's purpose was singular, to log people on/off. Therefore a simple design was envisioned, as shown in Fig. 14. A small box containing the username and password fields in the centre of the screen.

3.3.2 Channel Page

Since all the channel page had to do was show the channels that a user had access to a centralised list was thought of, as shown in Fig. 14. The user would be able to add channels using this list.

3.3.3 Scene Page

The scene page needed to display all the scenes in a channel. The interface in Fig. 15. was changed so that the map did not have the list to the side of it. This was so that users with smaller displays would still be able to use

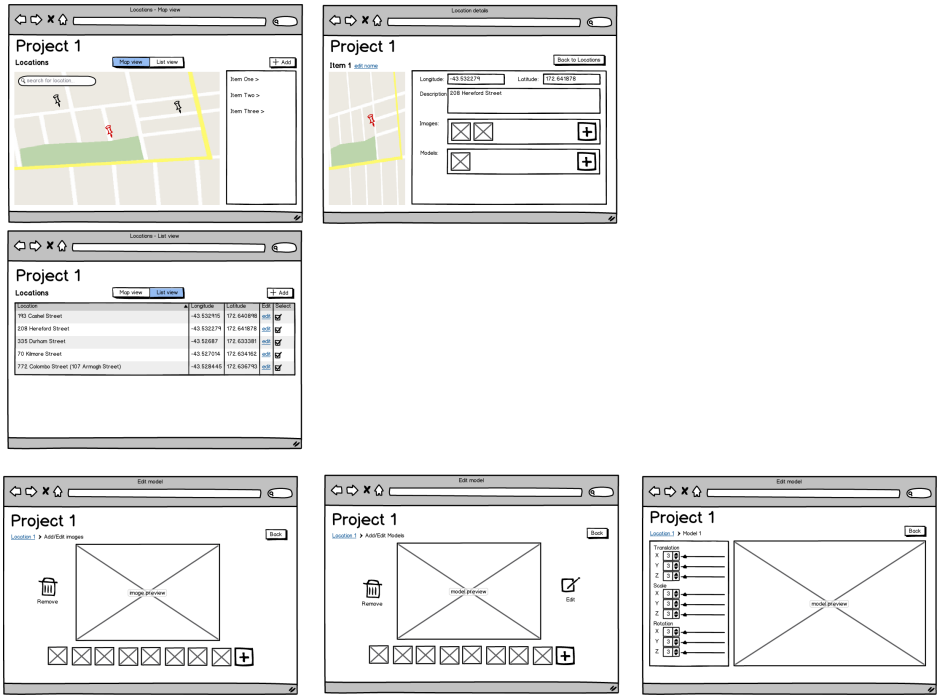


Figure 15: OutdoorAR authoring UI mock-up. Top two rows are scene page, bottom row is the model page

the map interface. As there was no longer a list on the screen it was decided that the list should be placed below the map. This meant that both list and map would be viewable at the same time by the user.

3.3.4 Model Page

The model page, shown in Fig. 15, was approached in the same way as the scene page. The location of the models would be displayed on the map so that the user had feedback on where the model was. The list below would also show more information on the model such as location, scale, and rotation.

4 Implementation

In the previous chapter we described the design process we went through and the final prototype designs that we wanted to build. In this chapter we describe the implementation of these designs starting with the client/server architecture, then the web based authoring tool as shown in Fig. 14. The mobile AR client was not developed over the course of this thesis as the project was given to another developer. The current CityViewAR client was able to display the information created using the web interface developed.

4.1 REST Client

In developing a Client/Server architecture one of the most important aspects was to ensure good data communication between the mobile or desktop Client and the Server. To facilitate good Client/Server communications the REST [3] architectural style was chosen. Since its inception in 2000 this has become a predominant web API design model. The REST architecture has key components that are required for the client-server systems such as:

- Scalability. As more items are added to the server the interactions don't change.
- Portability. The server is not concerned with the interface, just as the client is not concerned with the data storage. This allows developers to come and build custom interfaces for their own use.
- Error handling. The client and server can easily handle any error that occurs without the user being informed.

The REST API creates a client/server architecture that works using custom URLs from the client to retrieve and place information on the server. In addition to the URLs the REST API uses a set of request methods such as:

- GET This request retrieves the relevant information from the server.

- POST This adds data to the server in the relevant location.
- PUT This command updates the data on the server
- DELETE This deletes the relevant information

REST was used in a standard way over the majority of the site to retrieve, update, and add information to the server. Apart from this there were two cases where it was used differently:

1. When the user loaded the login page REST was used to check if they were logged in or not. On a success it would instruct the page to show a continue button. If it returned a failure the login dialogue would be shown, see Fig. 16.
2. If the user uploaded a model on the Model Page it needed to be linked to a scene. This was done if a success was returned from the uploading using a PUT command.

Figure 16: Login Dialogue

As an example of the type of REST code that needed to be developed, Listing 1. shows a block of code used to logout the user. The website first checks if a user is logged in by checking the session using a GET command.

If the user is logged in as indicated by a success command from the server the website will send a DELETE command which tells the server to log the user out.

```
1  $.ajax({
2    url:"http://132.181.247.31/outdoorar/ws/session/" + "?
         callback=?",
3    type:"GET",
4    crossDomain: true,
5    processdata:false,
6    success:function(data, textStatus, request)
7    {
8      $.ajax({
9        url:"http://132.181.247.31/outdoorar/ws/session/" +
             "?callback=?",
10       type:"DELETE",
11       crossDomain: true,
12       processdata:false,
13       success:function(data, textStatus, request)
14       {
15         checkLogin();
16       }
17     });
18   }
19 });
```

Listing 1: "Logout code block"

4.2 Client-Server Communications

jQuery [18] was the primary way that sever-client communications were performed. This was due to to it's popularity and it's inbuilt ability to handle REST commands. Using the jQuery ajax command, as shown in Listing 1., it encapsulated all the data and the REST instructions to the server. This made server-client communications highly standardised. The ajax command has several fields which are:

- url. The URL that the client is communicating to determined by the REST architecture.
- type. The REST command being communicated.
- crossDomain. This was used mostly for testing so that instructions could be sent from the same network.
- processData. This was set to false so that the data was not converted.
- data. The data being sent to the server.
- success. This allows a function to be run should the ajax command succeed.
- failure. Like success it allows a function to be run if ajax fails.

4.3 Web Interface

A web-based user interface was developed for adding content to the AR scenes hosted on the server. The interface was based around Google maps with HTML to add user interface components and Javascript to add interactivity. In this section we first describe the overall user interface and functionality and then how Google maps and Javascript was used to create the functionality.

Based on our the design exercise we did, the overall functionality that we wanted to support in the interface was following:

- Creation of Channels.
- Creation of Scenes.
- Uploading of Models.
- Specific placement of the Models.

Figures 17, 18, 19, and 20 show how this functionality was presented to the user through the user interface. First, when the user visited the web site they were presented with a login page that allowed them to login to the server with their own user name and password (see Fig. 17). Once logged in the user was presented with a list of AR channels they could add content to (see Fig. 18). They can pick the channel that they are interested in or create a new channel. Once a channel has been selected users can view the AR scene page (see Fig. 19) to select locations where they want to add content. From the scene page the user is taken the model page (see Fig. 20) to view and edit individual content elements.

The login page, Fig. 17, has fields for both the username and password of the user. jQuery and REST are used to check the status of the user and log them in.

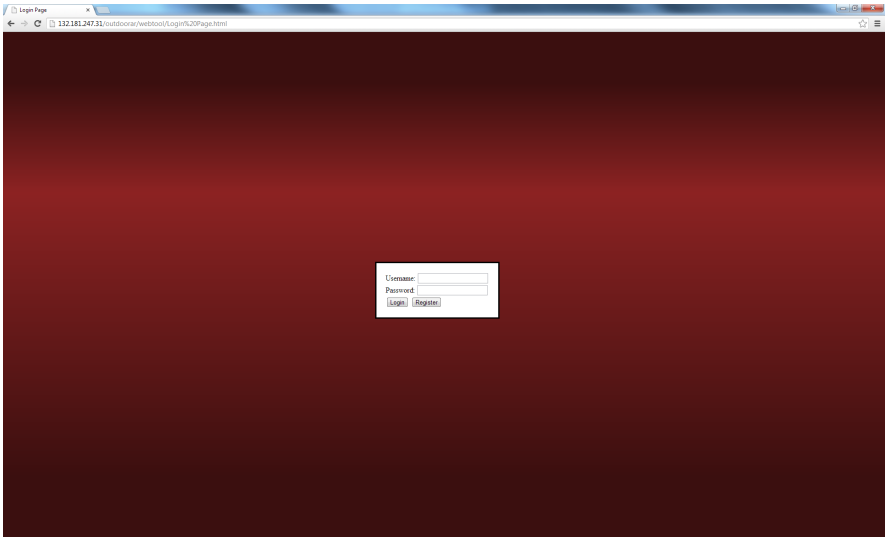


Figure 17: Final version of the Login Page

The channel page, Fig. 18, shows a table of the channels that the user has access to. This information is retrieved using jQuery and REST. The user can create a new channel which will be added to the server. The user

creates the channel by pressing the add channel button at the top of the table. A dialogue is then shown allowing the user to specify a name and category. The user then presses save and the channel is created. The user can view the scenes of a channel by either clicking on the table row or by pressing the button in the edit box of the row.

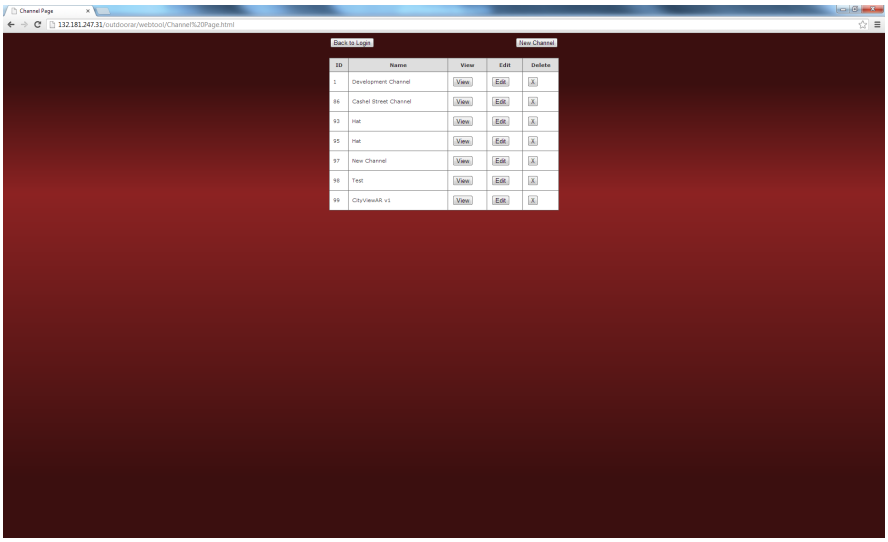


Figure 18: Final version of the Channel Page

The scene page, Fig. 19, shows a map with the location of all the scenes. A table underneath the map shows an alternate view of the scenes. The information shown is retrieved using jQuery and REST and is placed on the map using the Google maps API. The user can create a new scene on the server by right-clicking on the map or using the button on the table interface.

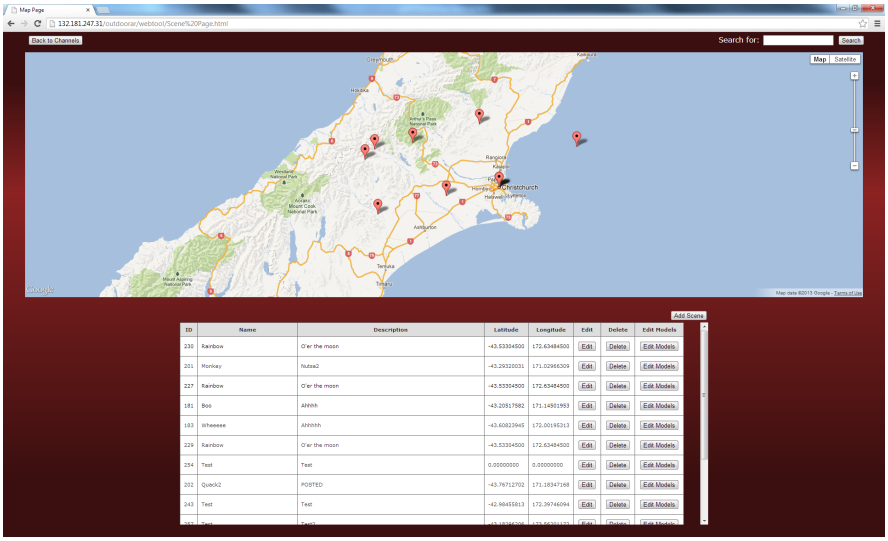


Figure 19: Final version of the Scene Page

The channel page, Fig. 20, shows a map with a central green marker which signifies the centre of the scene. The model locations are then placed around it relative to the scene centre. This is done by specifying the number of metres in the x, y, and z directions the that model is offset by. A table below the map shows information on the models offset, scale, and rotation. These values are set using a dialogue containing sliders and text boxes to specify the desired values. Models are uploaded to the server by browsing for the file and then uploading it.

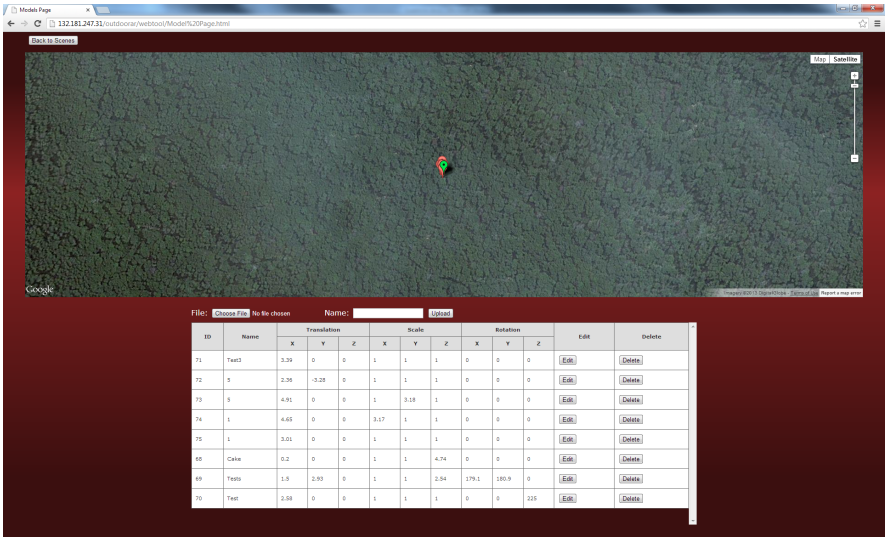


Figure 20: Final version of the Model Page

4.4 Use of Google Maps

Google maps [4] was chosen for the map components of the web interface due to its familiarity with end users and its large amount of documentation available. To use a custom Google map a developer licence is required, which allows for up to 25,000 uses of the map a day; more than enough for the prototype web interface.

Google maps allows for a lot of customisation of how its interface and the content on it is displayed. For the purposes of the web interface certain parts of the Google maps interface were removed/changed. This was achieved by using the code shown in Listing 2. This created the following initial settings:

- Default location is set to Christchurch.
- Default zoom is 8. This sets the maps to show the greater Christchurch area.
- Default map type was set to Road Map for scenes and Satellite for placing models.

- Pan controls were disabled. This is to show more of the map.
- Zoom and map type control were set in the top right of the screen. This was done so that controls were together.
- Scale control was removed. This was deemed unnecessary for the web interface.
- Street view and tilt was removed. This was so that a top down view of the map was always presented.

```
1  var myLatLng = new google.maps.LatLng
       (-43.423832,171.973859);
2  var mapOptions = {
3    zoom: 8,
4    panControl: false,
5    zoomControl: true,
6    zoomControlOptions: {
7      position: google.maps.ControlPosition.RIGHT_TOP
8    },
9      mapTypeControl: true,
10   mapTypeControlOptions: {
11     style: google.maps.MapTypeControlStyle.HORIZONTAL_BAR,
12     position: google.maps.ControlPosition.TOP_RIGHT
13   },
14   scaleControl: false,
15   streetViewControl: false,
16   overviewMapControl: false,
17   center: myLatLng,
18   mapTypeId: google.maps.MapTypeId.ROADMAP
19 }
20
21 map = new google.maps.Map(document.getElementById('
       map_canvas'),mapOptions);
22 map.setTilt(0);
23 overlay.setMap(map);
```

Listing 2: "Map initialisation code block"

These changes gave a maps interface without any clutter that would scale well to any screen size. Another feature used from Google Maps was the overlays. These allowed the map to return the latitude and longitude of a mouse event when clicked upon. These events were used to show context sensitive information when the map was clicked upon. For example, showing a list of options when the map was right-clicked, as shown in Fig. 21.

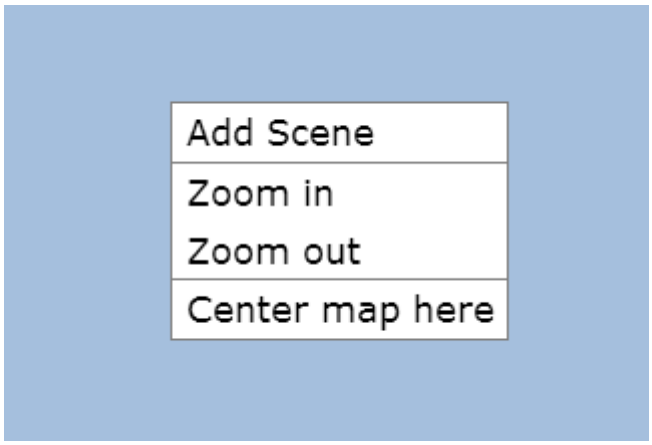


Figure 21: Right-click options

The polygon feature of the map was used to recreate models during the user evaluation. A polygon is a set of points that join together to make a shape. These were used to emulate the top down views of models being uploaded to the map. This was done by using the vertices of the model to create multiple polygons that were overlaid over each other to make the top down view. Fig. 22. shows a cube made from the vertices using the Polygon feature.

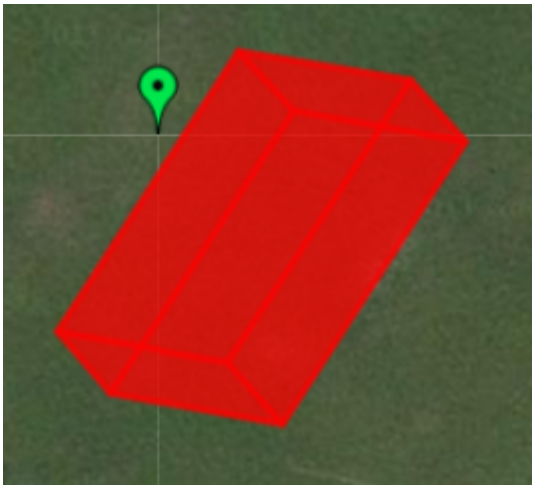


Figure 22: Using Polygons to make a cube

4.5 Use of Javascript

Most of the interactive functionality on the website was implemented through Javascript. HTML was used to create the user interface layout and Javascript was used to add interactivity. Javascript was chosen as it is very common and all modern browsers support it. It was used to perform all the user interaction that appeared on the web interface and performed all the calculations such as converting metres to latitude and longitude, as shown in Listing 3. The other sections were coded in Javascript and are broken down into their respective sections.

```
function calcNewLatLng(cLatLng, dx, dy)
{
  var cLat = cLatLng.lat();
  var cLng = cLatLng.lng();
  var r_earth = 6378*1000;
  var lat;
  var lng;
  if (cLat >= 0)
  {
    lat = cLat + (dy/r_earth)*(180/Math.PI);
  }
  else
  {
    lat = cLat - (dy/r_earth)*(180/Math.PI);
  }
  if (cLng >= 0)
  {
    lng = cLng + (dx/r_earth)*(180/Math.PI)/Math.cos(cLat
        *180/Math.PI);
  }
  else
  {
    lng = cLng - (dx/r_earth)*(180/Math.PI)/Math.cos(cLat
        *180/Math.PI);
  }
  return new google.maps.LatLng(lat,lng);
}
```

Listing 3: "Calculate offset latitude and longitude"

jQuery was also used to create UI elements. This is because the jQuery library allows for many UI elements that were used on the web interface. These elements come with many in built features and are easy to create. The two major ones used on the web interface were:

- Red highlighting when the user moves their mouse over a row in tables. This provides the user feedback that the row is click-able. An example is shown in Fig. 23.

- The dialogue used to manipulate model positions as shown in Fig. 24. jQuery was used to create the sliders and link them to the text boxes.

Figure 23: jQuery highlighting the row

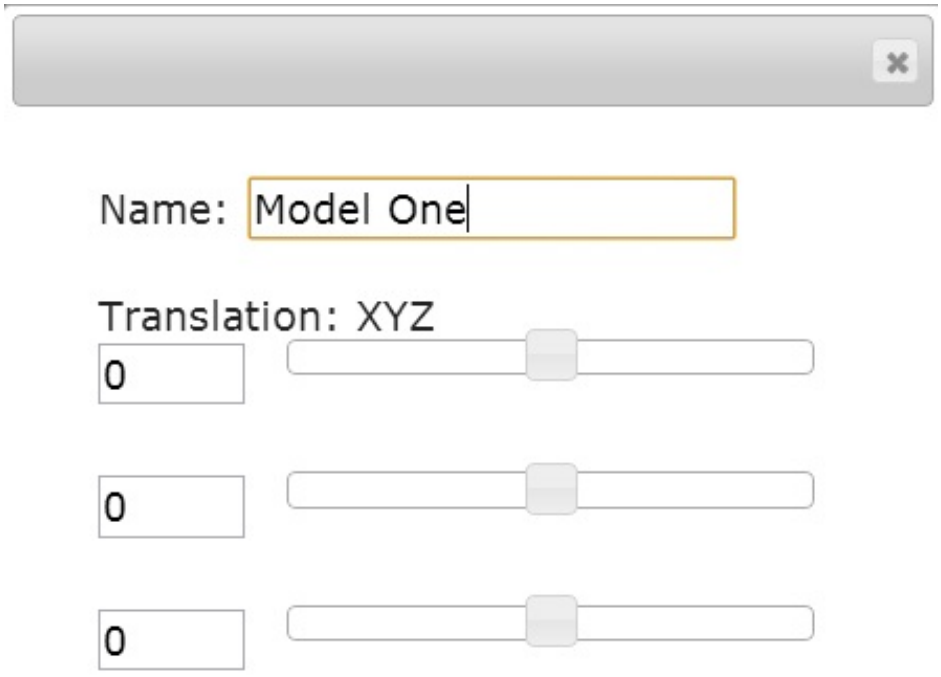


Figure 24: jQuery dialogue with sliders

Finally, jQuery had some useful features for use in the code. For example, the "each" command was used to sort the information coming from the server

into usable pieces. The code in Listing 4. was used to create the array of scenes that would be manipulated. It was also used to place a marker on the user interface map and add a scene to the table of available scenes.

```
1 $.each(temp, function(i, item){
2   var tempLatLng = new google.maps.LatLng(item.latitude,
        item.longitude);
3   scenes[item.id]=[item.name, item.description, item.
        categories, tempLatLng];
4   placeMarker(item.id);
5   addSceneTable(item.id);
6 });
```

Listing 4: "jQuery each example"

5 User Evaluation

5.1 Introduction

In the previous chapter we described the Client/Server architecture we implemented and the web interface used to adding content to the server and manipulate content already uploaded. In order to evaluate how intuitive the web interface was, and which interface style should be used, a simple user evaluation was conducted. In this chapter we report on the user evaluation and the results collected. The next chapter discusses lessons learned from the user evaluation.

5.2 Experimental Design

The overall goal of the User Evaluation was to see which web interface people preferred for adding the content to the server, and for uploading models. The two sessions were both counter-balanced using a Latin square to reduce learning effects.

The three conditions explored for adding content to the server were:

- Combined: A combined map/list interface that let the user choose which mode they wanted to use to add scenes, see Fig. 25.
- Map: A map interface that had users adding scenes by right clicking on the map, see Fig. 26.
- List: A list interface that required users to know all the information about the scene before adding it, see Fig. 27.

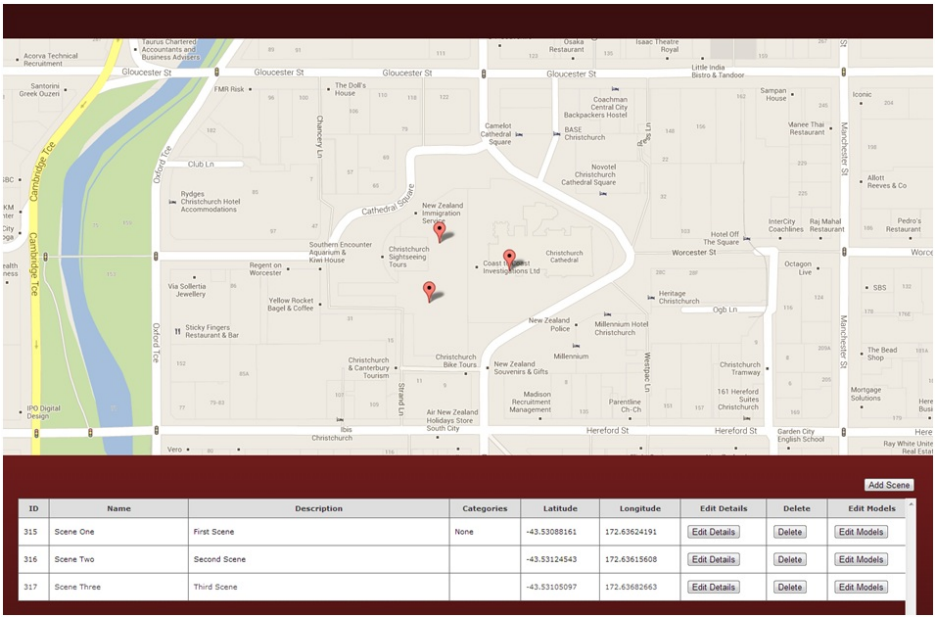


Figure 25: Condition One: Map/List combined interface

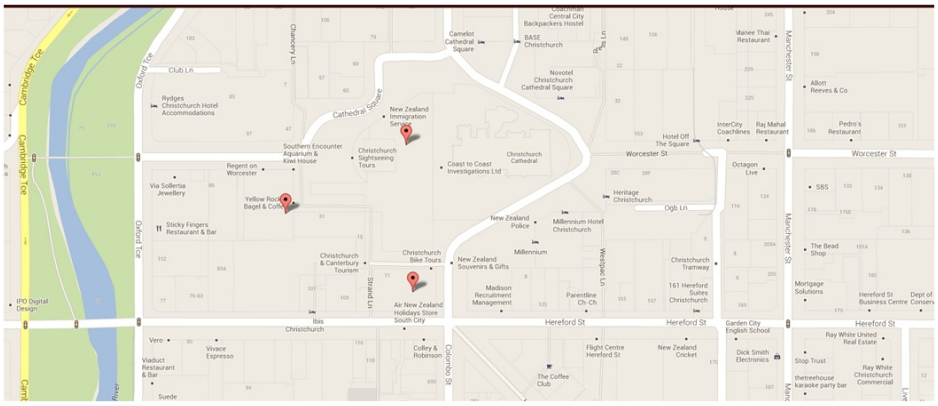


Figure 26: Condition Two: Map interface

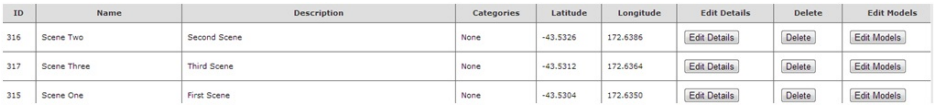
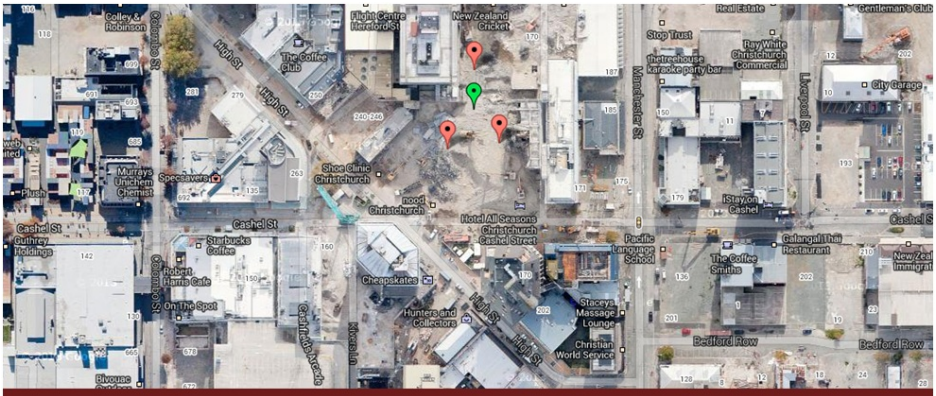


Figure 27: Condition Three: List interface

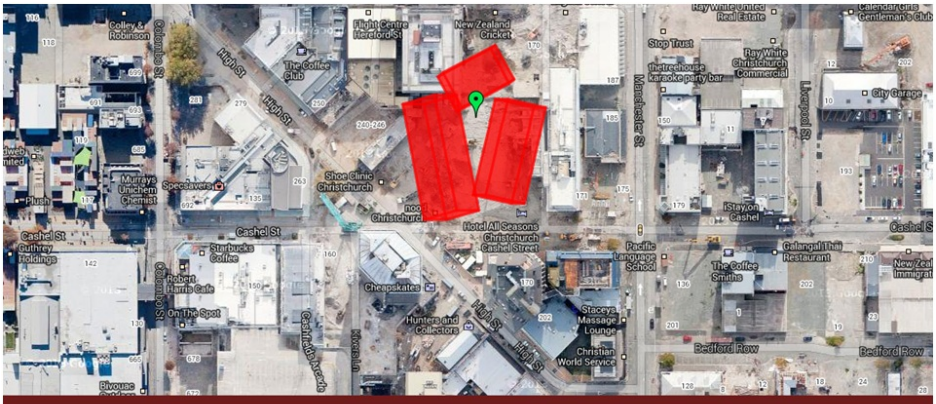
The two conditions explored for uploading content to the server were:

- Icon: An icon interface to show where the model was located, see Fig. 28.
- Footprint: A footprint interface that showed a top down view of the model, see Fig. 29.



ID	Name	Translation			Scale			Rotation			Edit	Delete
		X	Y	Z	X	Y	Z	X	Y	Z		
143	Model One	17	17	0	3	1	2.5	-107	84	0	Edit	Delete
144	Model Two	0	-20	0	3	1.5	1	-100	93	-41	Edit	Delete
145	Model Three	-18	20	-10	1	1	3	103	-56	0	Edit	Delete

Figure 28: Condition One: Icon interface



ID	Name	Translation			Scale			Rotation			Edit	Delete
		X	Y	Z	X	Y	Z	X	Y	Z		
0	Model One	17	17	0	3	1	2.5	-107	84	0	Edit	Delete
1	Model Two	0	-20	0	3	1.5	1	-100	93	-41	Edit	Delete
2	Model Three	-18	20	-10	1	1	3	103	-56	0	Edit	Delete

Figure 29: Condition One: Footprint interface

To do this the experiment was split into three sessions taking place over about half an hour as described below. A questionnaire was presented at the start of the session and at the end of each session. The first questionnaire asked basic demographic information about the participant. The questionnaires after a session dealt with what the participant thought of that session.

5.2.1 Session One: Training

This session was designed to introduce the user to the interface and train them on some of the functionality of the interface. The participant was seated in front of the computer and given the list of tasks to perform as follows:

1. Log into the site.
2. Create a channel called Channel One
3. View the channel

4. Create three scenes named Scene One to Scene Three centred around the University of Canterbury.
5. Move the scenes around slightly.
6. Press the Edit Models button for Scene One
7. Upload 2 models and name them Model One and Model Two
8. Move Model One 10 units in the X direction

These tasks guided the participant through the use of the website and the interactions needed to use it. After this session the users were given a Likert style questionnaire to gauge their thoughts on the website. This session typically took five minutes.

5.2.2 Session Two: Adding Scenes

This session took the participant through three different interface conditions for adding scenes to the server; the Map, List and Combined interfaces as described above.

During this session participants were asked to match a diagram (see Figs. 25, 26, and 27) given to them using the interface. They needed to upload and arrange content in the AR scene to match as closely as possible what was shown in the printed diagram. The time taken to match the diagram was measured during the session and after participants were asked to list in order of preference the interfaces and why they listed them in this order.

5.2.3 Session Three: Content Upload

This session was used to see which of two difference interfaces the participant preferred to upload models to the server; Icon or Footprint, as described above.

As before, the participants were asked to used the interface to match diagrams given to them (see Figs. 28 and 29). The time taken was recorded and participants were asked which interface they preferred and why.

5.3 Materials

The experiment was conducted in a 4 by 7 metre room. The materials used were:

- A desktop computer with two screens.
- Two chairs.
- A table.

The participant sat at the table with the desktop computer in front of them, as shown in Fig. 30. The experimenter, sat to the side of the participant to take notes and answer any questions they might have during the experiment. The primary screen of the computer was used to display the interface and questionnaire. The second, which was off to the side, displayed the instruction sheet for the participant to follow.

5.4 Participants

Before the experiment five individuals were used as a pilot study. These participants were given the interface to use. Their responses were recorded and used to improve the interface and following study design.

The experiment consisted of 12 individuals selected from the University of Canterbury between the ages of 20 and 35. This number ensured that each condition was completed twice. The participants were given a $5 voucher for partaking in the experiment.

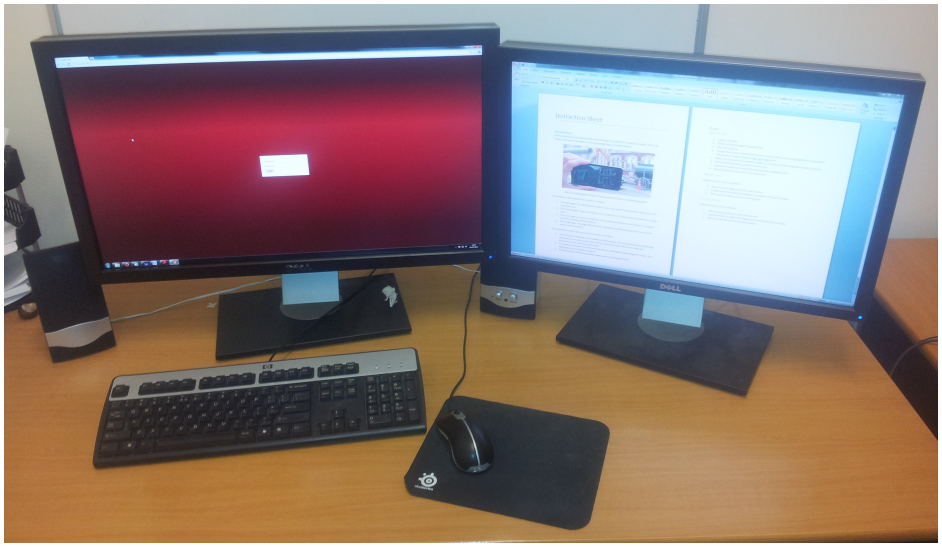


Figure 30: Experimental set-up

5.5 Results

This section will be split up into five parts:

1. A short overview of the Pilot study results as no metrics were recorded.
2. The demographic information of the participants.
3. The measurements taken from Session One.
4. The measurements taken from Session Two.
5. The measurements taken from Session Three.

5.5.1 Pilot Study

The pilot study found several flaws and bugs in the web interface that were corrected before the main experiment was performed. Examples of those that were found and fixed included:

- Fixing the interaction between buttons and the table row that they were on.

- Making the wording on the interface less ambiguous.
- Adding simple functionality such as hitting enter to submit.
- Fixing sliders that edited the models so that they moved.
- Fixing the building footprint display so that it was displayed correctly.
- Removing the ability to add non numeric numbers in a number field.

5.5.2 Demographics

Participant Age and Gender

As shown in Fig. 31. the participants ages were evenly spread between 21 and 33. Of the twelve participants, ten were male and two female.

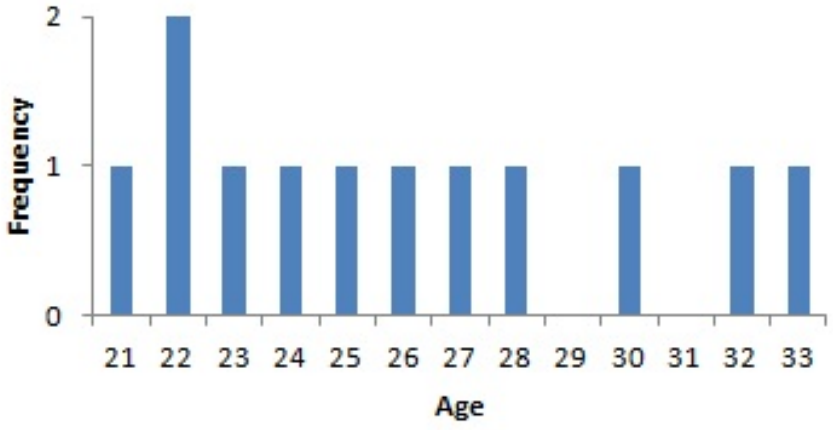


Figure 31: Ages of the participants

Participant Browsing Habits

The participants were asked how many hours per week they spent browsing the internet. All of them browsed the web at least 20 hours per week, meaning that they were all well versed in using the internet.

Participant Digital Map Experience The participants were asked to estimate how many times per week they used digital maps such as Google Maps. As shown in Fig. 32. the majority of participants were heavy users, using digital maps several times a week or more. Only two participants can be thought of as inexperienced with digital maps

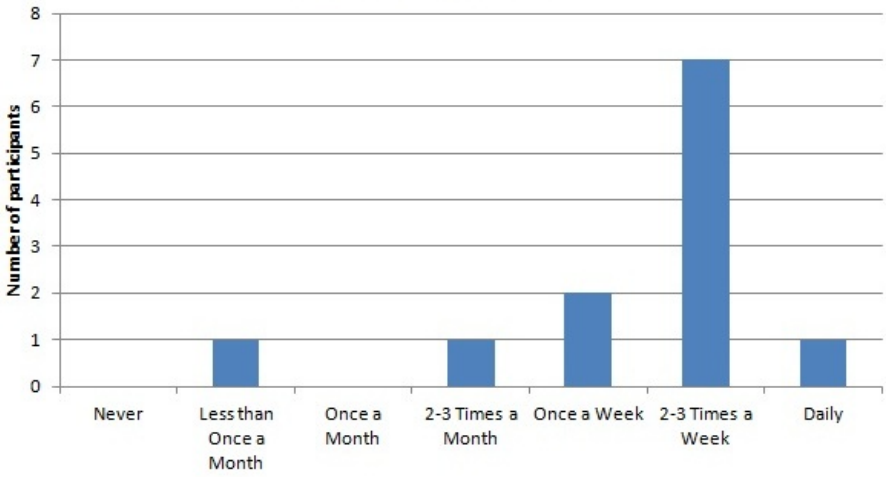


Figure 32: Usage of digital maps

5.5.3 Session One: Training

After being introduced to the interface the participants were given a questionnaire consisting of thirteen questions on a Likert Scale which ranged from 1 to 5 where 1 = Strongly Disagree and 5 = Strongly Agree. This was followed by a set of questions asking what the participants liked/disliked about the interface.

Q1: Overall, I am satisfied with how easy it was to use the interface

The mean, $\bar{x}$, for this question is 3.4167 with a standard deviation, σ, of 0.9962. The mean shows that participants found the interface easy to use. However the standard deviation shows that there was quite a variance with

these results. Fig. 33. shows a box plot of the results.

Q2: It was simple to use this system

The mean for this question is 3.5833 with a standard deviation of 0.7929. The mean of this question shows that participants found the interface simple to use. The standard deviation for this question indicates that the majority of the participants rated it between a 3 and 4. Fig. 33. shows a box plot of the results.

Q3: I could effectively complete the tasks using this interface

The mean for this question is 3.8333 with a standard deviation of 0.8348. These show that most participants were able to complete the tasks with a few outliers. Fig. 33. shows a box plot of the results displays this fact.

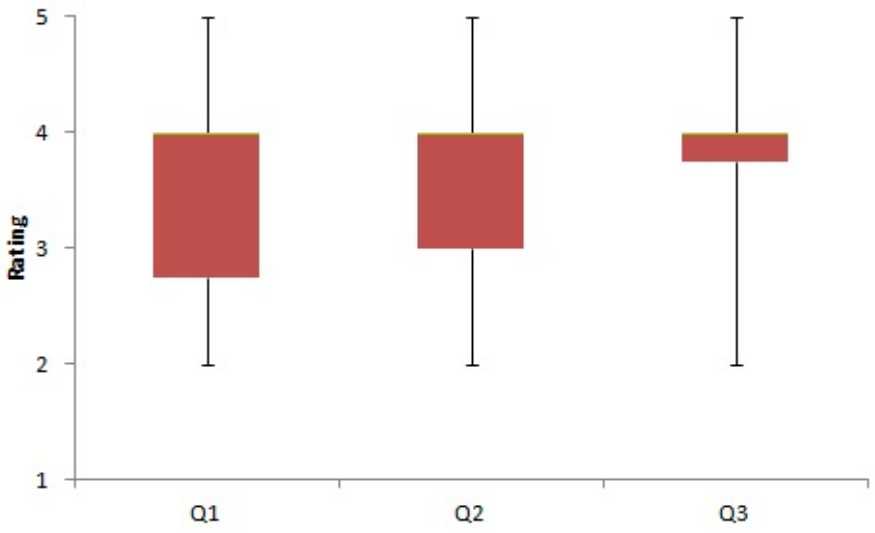


Figure 33: Questions one to three box plots

Q4: I felt comfortable using this interface

The mean for this question is 3.5000 with a standard deviation of 1.2431. These show that the participants had varying levels of comfort using the

interface. Fig. 34. shows a box plot of the results. The 1st quartile of the box plot is large compared to the rest of it showing that the participants that rated it lowly had quite different opinions.

Q5: It was easy to learn this interface

The mean for this question is 3.750 with a standard deviation of 1.2881. This shows that most people were able to learn the interface with an outlier that causes the large standard deviation. Fig. 34. shows a box plot of the results. It can be seen here that most users rated this question above a 3.

Q6: I believe I could become productive quickly using this system

The mean for this question is 3.6667 with a standard deviation of 1.1547. This shows that most participants were able to pick up the interface quickly, with a few participants struggling. Fig. 34. showing a box plot of the results with most participants rating over a 3.

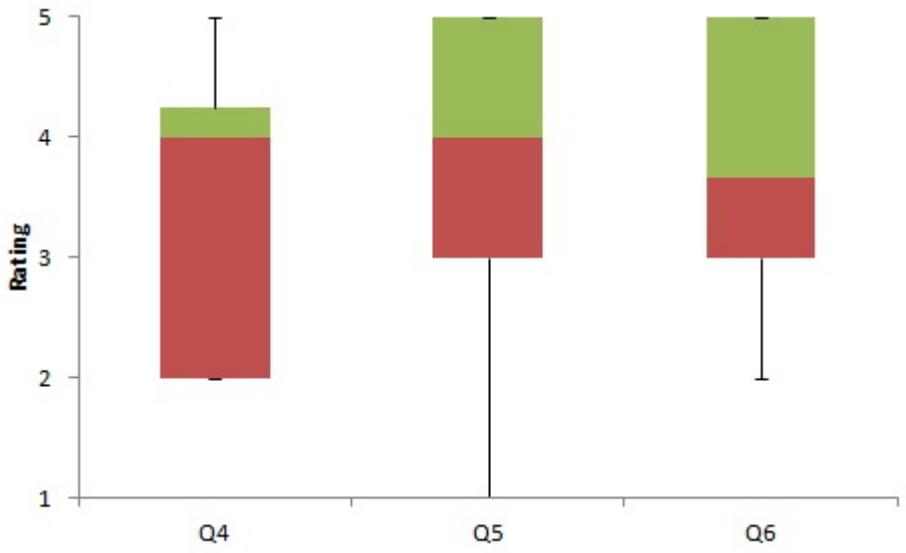


Figure 34: Questions four to six box plots

Q7: The interface of this system was pleasant

The mean for this question is 3.1667 with a standard deviation of 1.1146. The mean and standard deviation show that the responses were mostly neutral with a few participants rating highly or lowly. Fig. 35. shows a box plot of the results.

Q8: I liked using the interface

The mean for this question is 3.0833 with a standard deviation of 0.9962. This is a largely neutral response. Most users gave a rating of 2 to 4 with a single outlier at 5. Fig. 35. shows a box plot of the results.

Q9: Overall, I am satisfied with this interface

The mean for this question is 3.0000 with a standard deviation of 1.0445. This shows a neutral response to the question with a single participant giving it a rating of 5. Fig. 35. shows a box plot of the results.

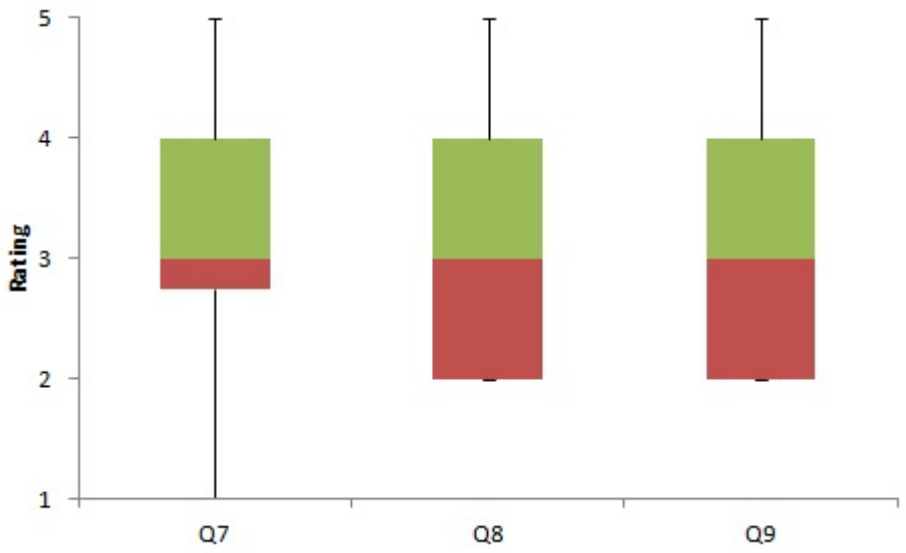


Figure 35: Questions seven to nine box plots

Q10: The interactions with the interface seemed natural

The mean for this question is 3.1667 with a standard deviation of 0.8348. This shows that most participants were neutral or thought that the interface seemed natural. Fig. 36. shows a box plot of the results. It can be seen that the lower outliers are lowering the median somewhat.

Q11: I was able to anticipate what would happen in response to the actions that I performed

The mean for this question is 3.3333 with a standard deviation of 0.9847. This result shows that most participants rated this question a 4. Fig. 36. shows a box plot of the results and the single large outlier.

Q12: It was easy to use the interface

The mean for this question is 3.9167 with a standard deviation of 0.9003. Most participants thought that the interface was easy to use as shown by this mean. Fig. 36. shows a box plot of the results with it's tightly clustered ratings.

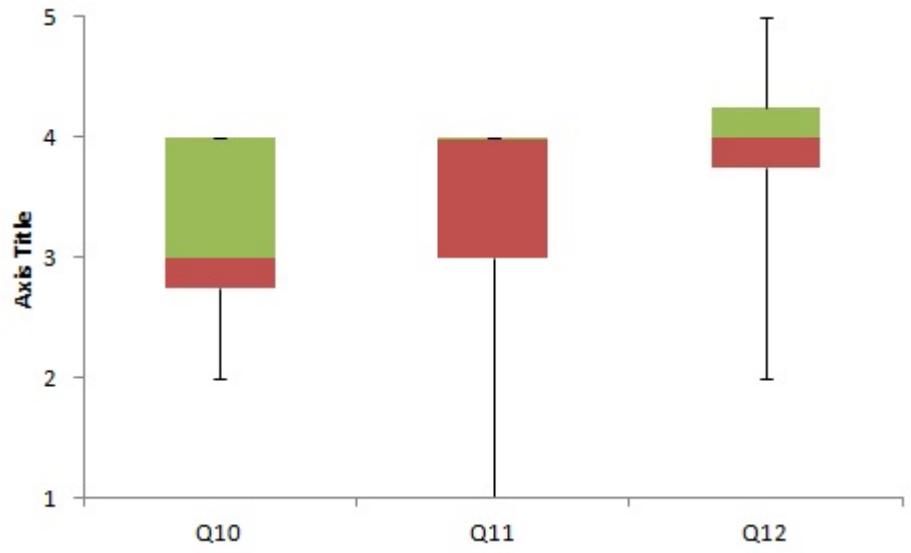


Figure 36: Questions ten to twelve box plots

Q13: I completed the tasks in a timely manner

The mean for this question is 3.9167 with a standard deviation of 0.6686. Most participants thought that they completed the tasks in a timely manner as shown by this mean. The low standard deviations shows that this was common across all participants. Fig. 37. shows a box plot of the results with the tightly clustered ratings.

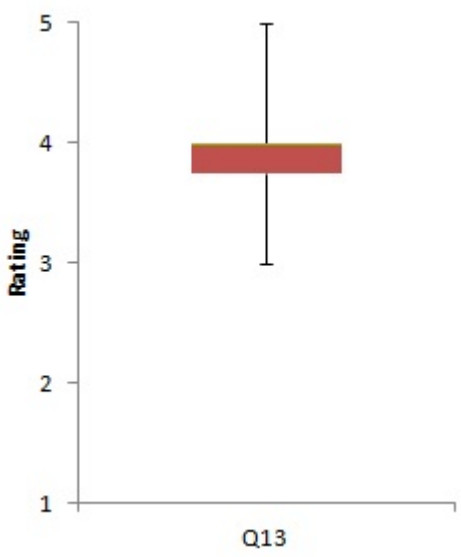


Figure 37: Question thirteen box plot

Q14: What did you like/dislike about the interface?

This qualitative question was asked to participants after they had completed the quantitative questions. Things that the participants like included:

- I liked that it was simple looking and not too much clutter.
- I liked the fact that you could click on the map for the latitude and longitude to appear.
- Dragging on the map.
- I could easy do the tasks with the interface.

- It seems like you use scene in different contexts.
- Placing markers and dragging them feels pretty natural.
- Placing scenes on a map was natural, especially drag and drop.
- Easy to use.

There was also the dislikes:

- Using co-ordinates on the list.
- Doesn't take advantage of modern techniques.
- The interface didn't look consistent.
- Box that enabled me to enter name of a scene on the map was not at the same point where I right-clicked.
- When translating the interface, I think it would be good to give more feedback on each label indicating which direction X, Y, Z is applying.
- Interface was very clunky with oddly placed buttons and the separation between a table and a map felt weird
- Some of the cues that I would have preferred for how to use some controls were not there.

This shows that even though the participants were able to use and interact with the interface it was not as natural as it could of been. The majority of the participants cited lack of visual cues and feedback as the most annoying aspect.

Q15: Did you encounter any problems?
Eight of the participants reported no problems with the interface. Those that had trouble found minor bugs in the code and small interface problems

that have since been fixed on the interface. The following problems were identified:

- Text on the buttons was different than the descriptions on the instructions.
- Marker disappeared.
- Buttons not quite clear

Q16: Any other comments?

This question was completely optional and only had two responses. Those were:

- Suggestion for bigger buttons.
- A sleepy participant made this clear.

5.5.4 Session Two: Adding Scenes

In this second session participants used three different interfaces to add scenes to the server. The time it took to add the scenes to the server was measured and participants were asked to rank the three interfaces for adding scenes to the server in order of preference. They were then asked to explain why they ordered the interfaces in this order.

Time Measurements

During the experiment completion times for the participants was taken. Table 1. and Fig. 38. show the time taken to complete each interface. A repeated measures ANOVA with Sphericity Assumed determined that mean time differed statistically significantly between interfaces $(F(2, 22) = 13.943$, $p < 0.05)$. Post hoc tests using Bonferroni correction revealed that the combined map/list interface did not have statistically significant difference to the list interface, $p = 0.1114$. However the map interface had statistically

significant difference to both the map/list combined interface, $p < 0.05$, and the list interface, $p < 0.05$. This can be viewed in Tables 2. and 3.

Table 1: Time taken on each interface

	Combined	Map	List
Mean	115.56	66.24	94.02
Std. Dev.	28.33	26.19	28.44

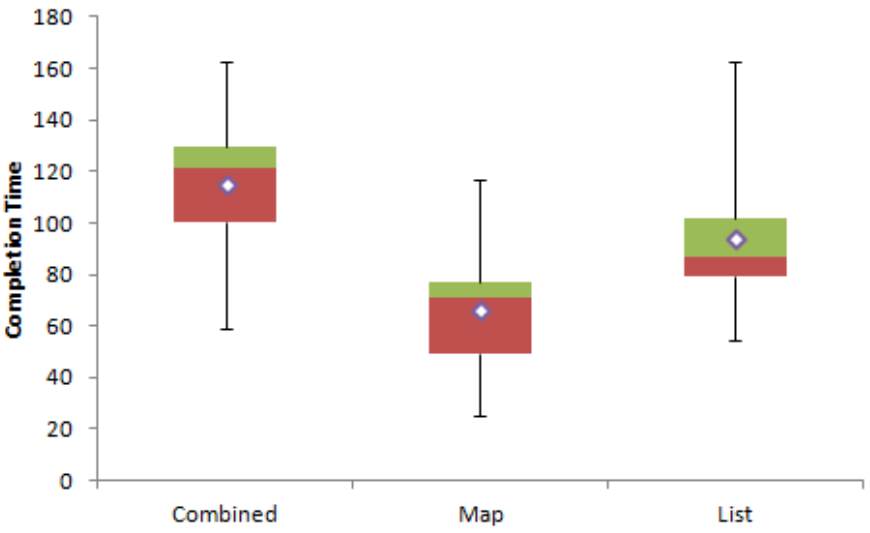


Figure 38: Box plot of time to complete. White dot shows mean

Table 2: Repeated ANOVA measurements with Sphericity Assumed

	df	F	p-value
Interface	2	13.9427	0.0001
Error	22		

Table 3: Pair-wise comparison results with Bonferroni correction

	Combined - List	Map - Combined	List - Map
p-value	0.1114	0.0037	0.0070

Following completing the task, users were asked to rank the interface in order of which one they preferred using.

Q1: Rank in order your preference for interface (1 being the best)

As shown in Table 4. and Fig. 39. two thirds of participants preferred the combined interface over the other alternatives with no one preferring the list over the others. The least popular was the list interface with 11 out of the 12 participants rating it their least favourite.

Table 4: Rating results

Rating	1st	2nd	3rd
List	0	1	11
Map	4	7	1
Combined	8	4	0

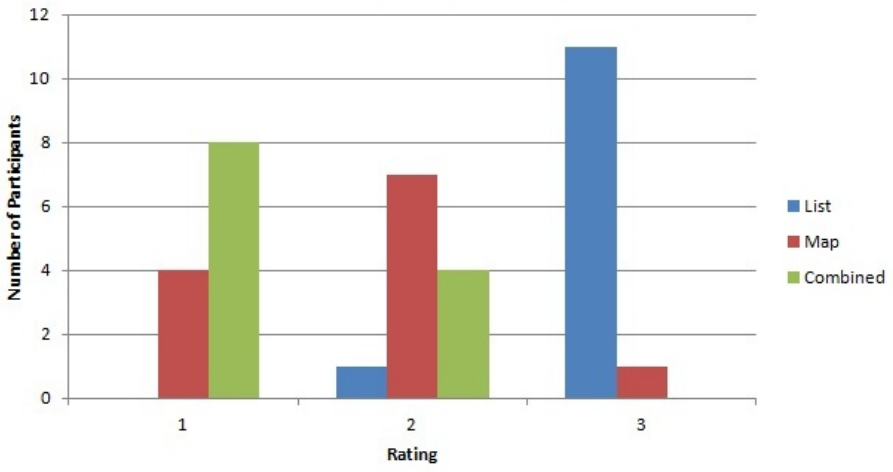


Figure 39: Rating of the interfaces

As shown in Table 5. there was a statistically significant difference in preferred interface, $\chi^2(2) = 16.1667$, $p < 0.05$. Post-hoc analysis with Wilcoxon signed rank tests was conducted with a Bonferroni correction applied, resulting in a significance level set at $p < 0.0167$, shown in Table 6. Median (IQR) preferred interface ranks for the list, map, and combined interface were 3, 2, and 1 respectively. There was a significant difference between the map and list interfaces, Z = -2.8099, $p < 0.0167$. There was no significant difference between the map and combined interfaces, Z = -1.2910, $p = 0.1967$. There was statistically significant difference between the list and combined interfaces, Z = -3.1530, $p < 0.0167$. This means that the list interface was ranked worst by the participants, but that they were split over which interface was the best between the Map and Combined interfaces.

Table 5: Friedman Test Results

N	12
Chi-square	16.1667
df	2
Asymp. Sig.	0.0003

Table 6: Wilcoxon Test Results

	Map - List	Combined - List	Combined - Map
Z	-2.8099	-1.2910	-3.1530
Asymp Sig. (2-tailed)	0.0050	0.1967	0.0016

Q2: Why did you like No. 1 the best?

The participants that preferred the combined interface gave the following reasons:

- I could work on the map, but see the details in the list.
- Because I could use both features. Depends on the situation one features might be more suitable than the other.

- Because it's complete.
- I got a direct feedback if everything is OK, and stored in the right way.
- Most tasks are easily accomplished using the map, but some (such as precise adjustment of latitude/longitude) are better done with a list that shows exact coordinates or values.
- Lets me choose whether I prefer accuracy (list) or ease of use (map).
- Because it was fun to use.

Overall the participants felt that the combined interface was best because it provided the best of both worlds allowing them to choose which one to use.

Participants preferred the map interface because:

- Because if you aren't aware of the coordinates you can just select using the map.
- Pretty simple.
- Most intuitive and easy to use - What you see is what you get!
- It was intuitive.

Overall the participants felt that the map interface was best because it was simple and fast to use. Participants clearly preferred having a map interface to work with when placing scenes.

Q3: Why did you think No.3 was the worst?

The participants that preferred the list interface the least gave these reasons:

- It was boring.
- It requires you to know all of the details with no pictorial view.

- Coordinates.
- Because entering data into a list is no fun. It's no challenge at all.
- Because you can't really what you do (you don't see where are the markers on the map).
- Nobody wants write on boxes if is possible avoid it.
- I associate coordinates with a position on a map. And maps are very easy for humans to "read".
- Took very long to type in coordinates.
- Because longitude and latitude are an extremely unintuitive way of thinking about objects on a map.
- Hard to use, but most accurate, so it depends on my priorities at the time.
- Because it gave me no visual information about the location.

The participants that liked the list lease did so for it's boring look and lack of visual feedback past the table itself. The participant that liked the map least had this opinion:

- Because, even thought it might be nicer to interact, if you want a lot of accuracy, it might be slower.

It is clear that the participants who liked the map least disliked the lack of visual feedback when interacting with the list interface.

Q4: Any other comments?

The optional comment question was answered by three participants and yielded these results:

- The only reason list/map is second is because it gives you a choice of which you wanted to use, otherwise it would be level with list as it can take longer to use as you have to decide how you want to input the information.
- Ditch the coordinate system.
- Can't use the "enter" key press

This section listed suggestions and small bugs that have since been taken into consideration/fixed.

5.5.5 Session Three: Content Upload

In this session the participants were asked to upload models to the server using two different interface types Icon or Footprint. The time taken to complete the upload task was measured for each condition, and the user asked to rank the two interfaces for uploading models to the server in order of preference. They were also asked to explain why they ordered the interfaces in this order.

Time Measurements

During the experiment completion times for the participants was taken. Table 7. ad Fig. 40. show the time taken to complete each interface. One participants results had to be left out due to problems with the recording. A paired t-test, shown in Table 8. was performed on the results, $t(10) = -2.3623$, $p < 0.05$. From this we can conclude there was a statically significant difference between the two interfaces.

Table 7: Time taken on each interface

	Icons	Footprint
Mean	121.33	195.08
Std. Dev.	76.37	70.93

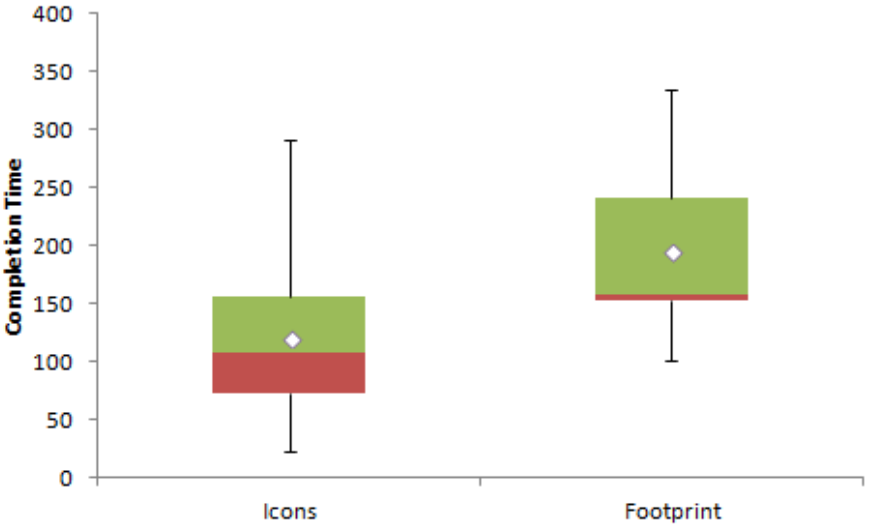


Figure 40: Box plot of time to complete. White dot shows mean

Following completing the task, users were asked to rank the interface in order of which one they preferred.

Q1: Rank in order your preference for interface (1 being the best)

As shown in Table 9. and Fig. 41. three quarters of participants preferred the footprint interface over the icon interface.

However, a Wilcoxon signed-rank test showed that there was no significance difference in preferred interface for uploading models to the server, Z = -1.1547, p = 0.2482, shown in Table 10.

Table 8: Paired Samples Test Results

t	df	p-value (2-tailed)
-2.3623	10	0.0398

Table 9: Rating results

Rating	1st	2nd
Footprint	8	4
Icon	4	8

Q2: Why did you like No. 1 the best?

The replies that participants that preferred the footprint interface included:

- Because you could see the area that it was covering pictorially instead of just having one point. Covers a range, therefore doesn't require you to know the exact point.
- Again: What you see is what you get: You immediately can tell the size of the model.
- Easier to understand.
- There are a recognisable geometric figure in the map.
- Because I got feedback for the scaling and position.
- Gave me an idea of the size of the model that I'm placing.
- Because it gave far more indication of what the model is doing.

Those that preferred the icons replied:

- So much less effort.
- Because I could match better with the model proposed and was more efficient.
- Simple to use.

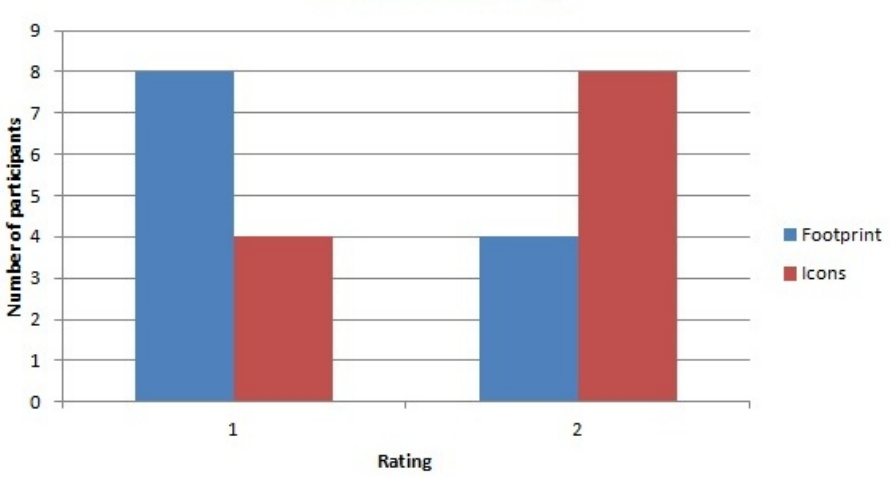


Figure 41: Rating of the interfaces

Table 10: Wilcoxon Test Results

	Footprint - Icons
Z	-1.1547
Asymp Sig. (2-tailed)	0.2482

- It was easy.

Overall, these replies show that participants liked the footprint interface for the feedback it gave regarding placement, scaling, and rotation of the model. Those that liked the icons preferred it for the speed and simplicity of placing the icons.

Q3: Why did you like No. 2 less?

The replies that participants that disliked the footprint interface included:

- Messing with dimensions.
- Because it was hard to match the model and it required time if I want to be precise.
- I did not dislike it. I just thought it was more complex. That is good - ONLY if necessary.

- It was harder.

Those that preferred the icons least replied:

- Not enough visual feedback.
- Didn't dislike it just liked the other better.
- No information about the extent of the model.
- Less precise.
- Position was OK. but it was not possible to see scale and rotate
- It wasn't bad, but it gave no indication of the scale and orientation of the model, only its position.
- Because it gave far less indication of what the model is doing.

Overall, these replies show that participants disliked the footprint interface for the time and complexity required to accurately place the models. Those that disliked the icon interface did so for its lack of accuracy and feedback. These results show that it came down to personal preference which interface the participants preferred.

Q4: Any other comments?
This question had two replies to it:

- Make the edges in an other colour, or the inner colour more transparent.
- Give the user the possibility to drag the objects on the map.

These replies show that the users wanted to be able to drag the models on the map rather than use the slider bars and have the edges of the model more defined. This revisions will be added in later versions.

6 Discussion

6.1 Demographics

The target audience for the web interface is city planners and AR developers. The participants in the study were drawn from the second category. We could definitely have used some more females in the study. It was heavily weighted towards males and we feel that adding some more females would have been a good idea. The age of the participants was a good spread across the age groups we were looking at, but maybe some older participants would have been good to see how those more unfamiliar with the internet would have handled the interface.

The participants were all heavy users of the internet. Maybe some more inexperienced users would have been good to include in the study to give it more of a spread. The maps usage was a good spread, with some participants being novice users of them and many being very experienced. We felt that most participants were moderate users and therefore didn't know everything that they could do with the maps. This gave us the opportunity to see how effective the cues we placed were.

6.2 Session One: Training

This session introduced the participant to the web interface. It was intended to show the user the interface and how it worked before testing the different interfaces. The Likert Scale questions all averaged between neutral and I agree (3 and 4). While it was good that they were all above neutral we would have liked them to average closer to 4. A larger sample size might have produced this. However we feel that running multiple usability studies during development might have been a better idea to get a more user friendly website.

Participants liked the ease of use with the web interface and it's simple

look. However there was some inconsistency with the interface and a lack of visual cues. The inconsistencies were things like the interface that was used to make a new channel being different from the one used to make a new scene. We should have all made these the same to stop confusion. The lack of visual cues such as obvious feedback that you can right-click on the map was a definite problem. These will need to add those on future versions of the web interface.

Several bugs that were not ironed out of the code were found during the user evaluation. A couple more usability studies during development would have been a good idea.

The users were given the freedom to ask questions about the interface during this session. Most participants asked questions about how to do things or needed to be guided when they looked confused about things. This shows that the interface is missing cues about what it can do. This will need to be addressed in future work.

6.3 Session Two: Adding Scenes

The time measurements shows the map to be the fastest, followed by list and then combined. The completion times had a statistically significant difference when taken as a whole. The map was likely fastest due to the simple nature of the interaction. Participants didn't need to be told how to quickly added locations and could work it out themselves. However when looked at against each other the combined and list interfaces were shown to have no statistically significant difference in completion time. This is likely because some participants would use the list in both circumstances.

This session tested which interface the participant preferred out of the map, list and combined interfaces. The map/list combined interface was the most popular of the interfaces as shown by the statistical significance.

However the preference between the map/list combined interface and the map interface was not. More participants in the study might have shown which interface was the most popular between the two. The list interface was likely less popular due to the tedious nature of its interaction. Participants did not like having to type out every detail to add a scene.

Participants liked having a map interface to work with, but some were put off by the list being included. We believe that the combined interface should be the interface that should continue to be developed. This gives users the option to play with the map or use the list when they already have a list of scene locations. As before the users managed to catch some bugs that made it through to the evaluation and make useful suggestions for future development.

From our own observations we can see that the web interface still needs some work. Participants were often confused with its use at first and needed to be guided before they were able to use it. This is not good interface design as we want users to be able to sit down and use it rather than needing to be taught.

6.4 Session Three: Content Uploading

The time measurements show that the icons were the fastest to complete. This difference in completion time was shown to be statically significant. We believe this is due to some participants skipping some of the placement. As the icons only showed their position and not rotation or scale we believe that some participants may of just moved the icons into the correct location and skipped the other dimensions.

This session tested which interface the participant preferred out of the icon and footprint interfaces. The footprint interface was rated as preferred the most, however statistical tests showed no statistically significant differ-

ence between the users preference of the interfaces.

Participants who liked the footprints liked it's accuracy, while those that liked the icons preferred it's speed. We believe this is because those who preferred the icons neglected the measurements that weren't to do with translation. A larger sample size would let us know for certain. Less bugs were found on this interface and useful suggestions were given.

6.5 Thoughts

We believe that any future development should use a combined interface when creating similar interfaces. This is so that information can be displayed in two formats for the user. Those that want to do it quickly can use the map and those with all the information needed can use the list. We think that a footprint interface is definitely the way to do. The ability to let the user see how the model is going to look gives the user vital feedback when placing models.

7 Conclusion/Future Work

In conclusion we can say that the web interface allows users to author content to be displayed on a mobile device. Users are able to: create channels, create scenes, and upload models to the server. The web interface is stable and in use with users around the HIT Lab NZ. Participants liked the interface and were able to complete the user study. However the participants had some valid suggestions on how to improve it.

Our main goals at the beginning of this work were:

1. To develop a client/server architecture for the CityViewAR application
2. To use this to extend the application to better support Urban Design.

A client/server architecture has been created over the course of this study. Using the REST architecture and by communicating with the other developers a client/server architecture for creating and loading content off the server has been created. Developers are able to easily modify their code to retrieve and create different items.

To support Urban Design the web interface has been created. This allows urban designers and developers to add new models to the server and retrieve them for viewing on a mobile AR application. It will allow them to precisely place models so that the end user will know exactly how the model will look.

Future work on the web interface will include:

- Implementing suggestions from the study participants.
- Fixing bugs in the code.
- Adding more functionality, such as uploading of other types of media.

Suggestions from users that we'd like to implement are:

- Add the ability to press the enter key to save data to the server. This is a relatively small change in the code.
- Making buttons bigger. At the moment buttons are a bit small and can easily be passed over. This is a simple fix.
- Letting users place models on the map by dragging them around. This is a large change in code, but should be implemented for user experience.

While performing future work more user studies will be performed. Expanding the interface are things like:

- Testing CityViewAR with the web interface to make sure that models show up.
- Research into simple authoring on the mobile device itself.

This will ensure that users will able to use the website in future and get a better experience as time goes on.

8 Bibliography

References

[1] BirdsView UG. BirdsView. `http://www.birdsview.de/`, 2011.

[2] Steven Feiner, Blair MacIntyre, Tobias Höllerer, and Anthony Webster. A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. *Personal Technologies*, 1(4):208–217, 1997.

[3] Roy Fielding. Representational state transfer. *Architectural Styles and the Design of Netowork-based Software Architecture*, pages 76–85, 2000.

[4] Google. Google Maps. `http://maps.google.co.nz/`.

[5] HITLabNZ. CityViewAR. `http://www.hitlabnz.org/index.php/products/cityviewar/`, December 2011.

[6] HOPPALA. HOPPALA Mobile Augmented Reality. `http://www.hoppala-agency.com/`, 2012.

[7] Volkert Jurgens, Andy Cockburn, and Mark Billinghurst. Depth cues for augmented reality stakeout. In *Proceedings of the 7th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: design centered HCI*, pages 117–124. ACM, 2006.

[8] Markus Kahari and David J." Murphy. MARA sensor based augmented reality system for mobile imaging device. `http://www.dirkreiners.com/ISMAR06Demos/Ismar06MARADemoProposal.pdf`, October 2006.

[9] MOB Labs. BuildAR. `http://www.buildar.com/`, 2009.

[10] Tobias Langlotz, Stefan Mooslechner, Stefanie Zollmann, Claus Degendorfer, Gerhard Reitmayr, and Dieter Schmalstieg. Sketching up the world: in situ authoring for mobile augmented reality. *Personal and ubiquitous computing*, 16(6):623–630, 2012.

[11] Tobias Langlotz, Daniel Wagner, Alessandro Mulloni, and Dieter Schmalstieg. Online creation of panoramic augmented reality annotations on mobile phones. *Pervasive Computing, IEEE*, 11(2):56–63, 2012.

[12] Layar U.S. Layar. `http://www.layar.com/`, 2012.

[13] Gun Lee, Andreas Dünser, Seungwon Kim, and Mark Billinghurst. CityViewAR: A Mobile Outdoor AR Application for City Visualization. In *11th IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2012) - Arts, Media, and Humanities Proceedings*, pages 57–64, Atlanta, Georgia, USA, 2012.

[14] metaio. Junaio. `http://www.junaio.com/`, 2006.

[15] Open Mobile Alliance. MobAR. `http://www.openmobilealliance.org/`, 2012.

[16] Wayne Piekarski and Bruce Thomas. Arquake: the outdoor augmented reality gaming system. *Communications of the ACM*, 45(1):36–38, 2002.

[17] Tagwhat Inc. Tagwhat. `http://www.tagwhat.com/`, 2012.

[18] The jQuery Foundation. jQuery. `http://jquery.com/`, 2005.

[19] Tonchidot Corporation. Sekai Camera. `http://sekaicamera.com/`, 2006.

[20] Wikitude GmbH. Wikitude. `http://www.wikitude.com/`, 2012.

9 Appendices

9.1 Appendix A: Consent Form

The following information and consent form were given to participants before the start of the study.

Human Interface Technology Laboratory New Zealand

University of Canterbury Private Bag 4800 Christchurch Telephone (64 3) 364 2349 Fax (64 3) 364 2095 Website www.hitlabnz.org

HITLabNZ

INFORMATION FORM

RESEARCH STUDY: Developing a Client/Server Architecture for a Mobile AR Urban Design Application

INVESTIGATORS: Michael Partridge, Prof Mark Billinghurst, Dr. Gun Lee, Dr Andreas Dünser.

You are invited to participate in the research paper entitled: Developing a Client/Server Architecture for a Mobile AR Urban Design Application

The aim of this project is to create a web interface to allow the uploading of media to the CityViewAR servers for viewing on the Mobile AR interface. This experiment will compare the interface developed to commercially available products.

Your participation in this experiment will have you perform a series of tasks across multiple interfaces. The tasks will include the uploading and positioning of media to each interface and the creation of new information on the interfaces. Between each interface you will be given a short questionnaire to rate your usage of each interface. After all interfaces have been used a final questionnaire will be given to rate your overall impression.

You may, at any time request to withdraw from the experiment for any reason with no consequence and your participation in the experiment will be terminated.

Upon the completion of your involvement in this study, we will also provide you with a $5 gift voucher.

The results of the project may be published, but you may be assured of the complete confidentiality of data gathered in this investigation: the identity of the participants will not be made public without their consent. To ensure anonymity and confidentiality, only the researchers will be allowed access to the video recordings of the participants. The recordings will be destroyed after a period of 5 years. They will be put in a secure and encrypted location that requires a password to gain access. The only individuals with the password will be the researchers in the study. Furthermore, the collected research data will also be kept in a secure and locked location. Only the researchers will have access to it via a key.

The project is being carried out as a research project by Michael Partridge, under the supervision of Prof. Mark Billinghurst and Dr. Gun Lee, who can be contacted by the following means. They will be pleased to discuss any concerns you may have about participation in the project.

Human Interface Technology Laboratory New Zealand

University of Canterbury Private Bag 4800 Christchurch Telephone (64 3) 364 2349 Fax (64 3) 364 2095 Website www.hitlabnz.org

HITLabNZ

Michael Partridge
HIT Lab NZ, University of Canterbury
Email: michael.partridge@pg.canterbury.ac.nz

Prof Mark Billinghurst
HIT Lab NZ, University of Canterbury
Email: mark.billinghurst@canterbury.ac.nz

Dr. Gun Lee
HIT Lab NZ, University of Canterbury
Email: gun.lee@hitlabnz.org

Dr Andreas Dünser.
HIT Lab NZ, University of Canterbury
Email: andreas.duenser@hitlabnz.org

This proposal has been reviewed and approved by the Human Interface Technology Laboratory, University of Canterbury and the University of Canterbury Human Ethics Committee Low Risk process.

Please take this form with you when you leave.

Human Interface Technology Laboratory New Zealand

University of Canterbury Private Bag 4800 Christchurch Telephone (64 3) 364 2349 Fax (64 3) 364 2095 Website www.hitlabnz.org

CONSENT FORM

RESEARCH STUDY: Developing a Client/Server Architecture for a Mobile AR Urban Design Application

INVESTIGATORS:

Michael Partridge

HIT Lab NZ, University of Canterbury

Email: michael.partridge@pg.canterbury.ac.nz

Prof. Mark Billinghurst

HIT Lab NZ, University of Canterbury

Email: mark.billinghurst@canterbury.ac.nz

Dr. Gun Lee

HIT Lab NZ, University of Canterbury

Email: gun.lee@hitlabnz.org

Dr Andreas Dünser.

HIT Lab NZ, University of Canterbury

Email: andreas.duenser@hitlabnz.org

I have read and understood the description of the above-named project. On this basis I agree to participate voluntarily as a subject in the project, and I consent to publication of the results of the project with the understanding that anonymity will be preserved. The data will be kept for up to 5 years before being destroyed.

I understand that I will be recorded during the experiment, but the recording will only be viewed by researchers directly associated with the project. I also understand that the recording will be kept for up to 5 years before being destroyed.

I understand also that I may at any time withdraw from the project, including withdrawal of any information I have provided.

I note that the project has been reviewed ***and approved*** by the University of Canterbury Human Ethics Committee.

_____	_____	_____
Participant (Print name)	Signature	Date

9.2 Appendix B: Questionnaire

Participants were asked to fill in the following questionnaire as the study progressed.

User Study Questionnaire

Age: _____ ☐ Male / ☐ Female

How many hours do you spend web browsing per week?

0-9 10-19 20-29 30-39 40+

How often do you use a digital map (e.g Google Maps)?

Every day 2-3 times a week 2-3 times a month Once a month Not at all

Session One

Experience Questionnaire	Strongly Disagree 1	Disagree 2	Neutral 3	Agree 4	Strongly Agree 5
Overall, I am satisfied with how easy it is to use the interface	☐	☐	☐	☐	☐
It was simple to use this system	☐	☐	☐	☐	☐
I could effectively complete the tasks using this interface	☐	☐	☐	☐	☐
I felt comfortable using this interface	☐	☐	☐	☐	☐
It was easy to learn this interface	☐	☐	☐	☐	☐
I believe I could become productive quickly using this system	☐	☐	☐	☐	☐
The interface of this system was pleasant	☐	☐	☐	☐	☐
I liked using the interface	☐	☐	☐	☐	☐
Overall, I am satisfied with this interface	☐	☐	☐	☐	☐
The interactions with the interface seemed natural	☐	☐	☐	☐	☐
I was able to anticipate what would happen in response to the actions that I performed	☐	☐	☐	☐	☐
It was easy to use the interface	☐	☐	☐	☐	☐
Du	☐	☐	☐	☐	☐

Why did you like about the web interface?

Did you encounter any problems?

Any other comments?

Session Two

Condition Order: _____

Rank in order from favourite to least favourite the interfaces?

1.

2.

3.

Why did you like No 1. the best?

Why did you dislike No. 3?

Any other comments?

Session Three

Condition Order: _____

Which interface did you like the best?

Why did you like it?

Why didn't you like the other one?

Any other comments?